1995

Maple Tools For Hydrodynamic Interaction Problems

Christopher Pratibha

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

Canada

MAPLE TOOLS FOR HYDRODYNAMIC INTERACTION PROBLEMS

by

Pratibha

Department of Applied Mathematics

Submitted in the partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

Faculty of Graduate Studies
The University of Western Ontario
London, Ontario
July 1995

Chapter 1

Introduction

"Where shall I begin, please your Majesty?" he asked.
"Begin at the beginning," the King said, gravely, "and go on till you come to the end: then stop."

Lewis Carroll, *Alice's Adventures in Wonderland.*

1.1 Suspension of particles

Electrorheological (ER) fluids are the fluids whose flow properties change when they are exposed to an electric field. ER fluids are made by suspending electrically susceptible particles in fluid. When an external electric field is applied to the system, the viscosity of the fluid can change by an order of magnitude. These fluids are being studied in particular by the automotive industry with a view to creating new vibration-control devices, clutch mechanisms, etc. To date there has been considerable frustration within industrial research laboratories because of the slow translation of the apparent possibilities into practical devices. A recent study by Hartsock *et al* (1991), for the Ford motor company, determined that at present engine mountings were the only devices that might feasibly be made using ER fluids,

ABSTRACT

Several calculations of interactions between particles suspended in fluid are first described. In order to complete these calculations efficiently, several facilities that the present version of Maple does not provide are identified. These include the efficient computation of series solutions of ordinary differential equations, the management of large expressions and the convenient solution of polynomial equations in terms of a set of algebraic extensions dependent over the rationals. These facilities are developed and the calculations carried through to completion. A feature of the computations is the solution of ordinary differential equations to high accuracy by series methods. The mathematical and programming development of Maple packages to solve polynomial equations, and to obtain many terms of series solutions to ordinary differential equations are described. The series-solution package applies to linear ordinary differential equations of arbitrary order about ordinary points and regular singular points.

ACKNOWLEDGMENTS

First of all, I would like to sincerely thank Prof.D.J.Jeffrey for his continuous guidance, cooperation and patience during the course of this thesis. His keen interest in this research contributed a great deal to the completion of this thesis.

I would also like to thank Dr.R.M.Corless, whose course helped me a great deal in making a good background for computer algebra systems.

Thanks to all colleagues and staff of the Dept. of Applied Maths at UWO who made my stay in Canada pleasant and comfortable.

Last but not least, I would like to thank Kamal and Ankit, for giving me moral support and encouragement throughout this work.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

"Where shall I begin, please your Majesty?" he asked.
"Begin at the beginning," the King said, gravely, "and go on till you come to the end: then stop."

Lewis Carroll, *Alice's Adventures in Wonderland.*

1.1 Suspension of particles

Electrorheological (ER) fluids are the fluids whose flow properties change when they are exposed to an electric field. ER fluids are made by suspending electrically susceptible particles in fluid. When an external electric field is applied to the system, the viscosity of the fluid can change by an order of magnitude. These fluids are being studied in particular by the automotive industry with a view to creating new vibration-control devices, clutch mechanisms, etc. To date there has been considerable frustration within industrial research laboratories because of the slow translation of the apparent possibilities into practical devices. A recent study by Hartsock *et al* (1991), for the Ford motor company, determined that at present engine mountings were the only devices that might feasibly be made using ER fluids,

but shock absorbers would become feasible within the medium term, if research progress could be maintained.

ER effects were reported about a century ago by Duff (1896). He observed small variations in viscosity of glycerin and castor oil by the application of electric fields. After a gap of 40 years, work on electroviscous effects resumed under Björnståhl & Snellman in the mid to late 1930's (Björnståhl 1935; Björnståhl & Snellman 1937, 1939) and da Andrade & co-workers afterwards (da Andrade & Dodd 1939, 1946, 1951; da Andrade & Hart 1954). Björnståhl and Snellman studied the electroviscous properties of selected polar and non polar liquids and colloidal dispersions of metals and sulfur in dielectric liquids and reported moderate changes in electroviscosity. da Andrade attributed the apparent increase in viscosity to orientation of polar molecules and formation of ion clusters at the electrodes. Owing to its potential engineering applications, the research in electrorheology attracted a lot of interest after Winslow (1949) recognized the electrically induced fibration of small particles in fluid liquid suspension. Since then, the research in this field has diversified in two directions, studies of the properties of ER fluids by the experimentalists in order to manufacture a suitable ER fluid, and theoretical studies by applied mathematicians and physicists.

The theoretical studies, which at this time are still in their early stages, are to investigate systems, such as suspensions, that show strong departures from the simple Newtonian laws of fluid flow. The first calculation of the effective flow properties of a suspension in terms of its constituents is usually taken to be Einstein's (Einstein 1956). For a comprehensive review of this topic, the reader is referred to

Batchelor (1974) and Jeffrey & Acrivos (1976).

The importance, for this thesis, is that theoretical approaches are all based in the idea of interactions between particles. The term 'interaction' refers to the relations that exist between properties that can be considered as possessed by the particle as a whole and parameters describing its environment. Examples of such particle properties include the location of its center of mass, the center of mass velocity, the rotation of a rigid body, and the average temperature, and in addition include less commonly known quantities such as the stresslet and the electric dipole strength. Examples of parameters describing the environment include the average velocity of fluid near the particle, the temperature gradient in the vicinity of the particle, the presence of other particles, and so on. In this thesis we shall consider specifically the calculation of the force and couple on a spherical particle near a plane wall, as well as the inviscid flow around a particle touching a plane wall.

Our particular interest lies in the development of general tools for solving these problems, as much as providing data for special cases. Symbolic manipulations can be used in such studies, but not before improvements are made to the existing tools in computer algebra systems. Some of the problems in existing tools, such as intermediate expression swell, large computational time and memory requirements are addressed in this thesis and suitable improvements are made. Current status of the available symbolic manipulation programs is given below.

1.2 Computer algebra systems

Computer algebra systems (CAS) are simply the programs which enable one to manipulate mathematical expressions symbolically. For most of the computer literates, the word *computing* means number crunching or numerical calculations. Manipulation of complex mathematical expressions is considered to be a daunting task for computers. Before computers appeared on the scene, a calculation usually consisted of a mixture of numerical calculation and calculation by mathematical formulas or algebraic calculation. All the numerical calculations were preceded by a manipulation of algebraic formulas, if the work was to be within the bounds of what is humanly possible. In the 19th century, several large calculations have a substantial amount of formula manipulations.

Among the famous calculations was Le Verrier's calculation of the orbit of Neptune, which started from the disturbances of the orbit of Uranus, and led to the discovery of Neptune. The most impressive and probably the largest calculation with pencil and paper is by the French astronomer Charles Delaunay (1867). He took 10 years to calculate the orbit of the moon as a function of time, and another 10 years to check it. If only he had a computer with algebraic manipulation capabilities.

It is for such large and complicated algebraic expressions, scientists developed such systems which can share the burden of doing algebra in simplifying the mathematical expressions. A complete computer algebra system (CAS) is desired to have the following:

- not restricted to a particular application

- good user interface
- wide range of built-in functions
- interactive user interface
- portable to a variety of computers
- good graphical capabilities

1.3 Need for CAS

Computer algebra can save both time and effort in solving a wide range of problems. In general, the solution obtained through CAS is much more accurate than the traditional techniques. A lot of problems, which were stopped in the past due to their huge size, can be solved using these systems.

Algebraic solutions are always exact as opposed to their numerical counterparts. Errors arising from rounding and truncation of numerical quantities in the intermediate steps make the final solution approximate and sometimes unacceptable. On the other hand, the accuracy in CAS is controlled by the software, therefore the user has total command over how accurate results are needed.

Algebraic solutions presents the final results in a reasonable form for humans to understand. An algebraic solution can indicate far more about the relationship between the variables in expression than a host of figures. A undergraduate class was asked to fill the blank in following two questions.

1. A circle of radius 2 has a $\cdots\cdots\cdots$ of 12.56637061.

2. A circle of radius r has a ..... of $2\pi r$.

No one could provide the answer to the first problem, whereas almost everyone wrote **perimeter** in the second problem. This simple example explains how important is to have a solution in the algebraic form.

The availability of CAS on a wide range of platforms encourages researchers to investigate larger problems than they could ever attempt using traditional methods. Even for numerical solutions, it is always wise to simplify the solution as much as possible through a CAS. This will result in making less numerical errors in intermediate steps.

In the next section, a brief history of CAS is presented. More detailed discussion can be found in several recent texts (Harper *et al* 1991; Geddes, Czapor and Labahn 1992; Davenport *et al* 1993).

1.4 Review of CAS

The origins of the discipline of computer algebra can be found in Isaac Newton's *Arithmatica Universalis* (1728), where methods for manipulating universal mathematical expressions (*i.e.* formulas containing symbolic indeterminates) and algorithms for solving equations built with these expressions are systematically discussed. The concept of carrying out algebraic manipulations using computers is more than a century old, when Augusta Ada Byron pointed out in 1844, that computer could

arrange and combine its numerical quantities exactly as if they were letters or any other general symbols; and in fact it might bring out its results in algebraic notation, were provisions made accordingly.

However, the computer algebra could not be practised until Kahrimanian (1953) and Nolan (1953) published their M.A. theses regarding analytic differentiation on a digital computer. It wasn't until the development of LISP in 1960-1961, a laaguage for list processing, that the algebraic computation could become more popular among scientists. Several CAS were later written in LISP.

J.Slagle of M.I.T. wrote a program in 1961 to carry out indefinite integration. This program, written in LISP, was called SAINT for Symbolic Automatic INTegration and was based on a number of heuristics. Later, in 1966-1967, J. Moses developed another program SIN (Symbolic INtegrator) and ironically, SIN was more efficient than SAINT, because of its algorithmic approach.

During 1962-1964, Sammet, Tobey and others at IBM developed a computer algebra system FORMAC. This system was designed for the manipulation of elementary functions and polynomials. Around the same time, W.S.Brown and others at Bell laboratories also designed a system ALPAK. Both of these systems were closely related to FORTRAN.

Now scientists felt the need of a computer algebra system, which can be used interactively. In 1965, Carl Engelman at M.I.T. came up with MATHLAB, a system for manipulating rational functions and polynomials. This program was written in LISP and was the first interactive system designed to be used as a symbolic

calculator.

REDUCE was the first general purpose algebraic package written by A.C.Hearn in 1968. It's biggest advantage was the portability to several platforms, which made it instantly popular. Later, in that decade, several general purpose systems were developed for algebraic computation, ALTRAN, SAC-I, CAMAL (**CAM**bridge **AL**gebra systems) to name a few. The latter was developed mainly for computations in celestial mechanics and general relativity (Barton & Fitch 1974).

During the period 1969-1982, J.Moses, W.Martin and others developed MACSYMA at M.I.T. For the first time, a CAS has included such computations as symbolic integration, limit calculations and the solution of equations. In the 70's, these systems were put into scientific applications. A very strong limitation of these systems was the high computer resource requirements. Except for muMATH (Stoutmeyer & Rich 1983), a package written for microcomputers, other packages were meant only for mainframe computers.

The increasing popularity of microcomputers, forced scientists to develop packages which are more efficient in terms of computer resources. This resulted in the development of CAS based on new *system implementation* languages, such as C (Mathematica, Maple, SMP), which can control the computer resources more flexibly and efficiently.

In 1980, K.O.Geddes and G.H.Gonnet initiated the development of Maple at the University of Waterloo [Char *et al* (1983)]. This was named Maple due

to its Canadian origin. Maple was designed with a modular structure: a small compiled kernel and a rather large number of routines to be interpreted by the kernel. The kernel included the facilities like integer and rational arithmetic, polynomial manipulation and efficient memory management.

SMP (Symbolic Manipulation Program) was also written in C by S.Wolfram and could be ported on a wide range of computers. It differed from other systems in that it used a rule based language interface. The latest computer algebra packages developed were Derive and Mathematica. Mathematica (Wolfram 1991) became quickly popular because of its excellent graphic capabilities. Mathematica had its own programming language which closely followed its predecessor SMP.

Derive is the successor of muMATH for personal computers. It has a friendly interface and nice plotting facilities. A number of improvements were also made to the existing packages. Several outside developers added new packages MACSYMA and REDUCE. AXIOM, a successor to Scratchpad was developed by Jenks & Suter (1992) at IBM. Axiom has a great advantage over other existing systems in that it allows users to write algorithms over general fields or domains, while other CAS develop algorithms for a specific domain only (field of rational numbers or domain of polynomials over the integers). AXIOM is now marketed by N.A.G. Inc.; it can overcome some limitations other systems share, and can be the prototype for the next generation of CAS.

1.5 Limitations of CAS

From the above discussion, it may seem like that the CAS are the answer to all our problems and the opportunities these systems offer are unlimited. But this is a false impression. All of the CAS are bound to have bugs and limitations.

A bug may be defined as something bad that a program does contrary to the programmer's intent. Stoutemyer (1991) discussed several bugs and limitations of CAS in a very interesting article. Heck (1993) also mentions some of the limitations of CAS.

- Computer algebra systems require a large amount of computer resources in terms of the computing time and memory. It is true that a CAS can be used to obtain any amount of precision, but there is a negative aspect to that. Since CAS uses software floating point arithmetic, one requires higher computer resources with the increase in precision.

- Another problem of CAS is the exponential rise in the size of intermediate expressions, even if the final result of the calculations is very small. This is well known to the practitioners of computer algebra as the *intermediate expression swell.*

- As such, there is a lack of compactness in all the CAS. In case of large expressions, it is often difficult to get a insight of the results. Apart from that, some times it even stops the calculation due to the exhaustion of the computer resources.

- No tool can be developed away from its area of application. Therefore by the

application of CAS to real problems from other branches of science, we shall see ways in which they need to be improved.

In this thesis, several techniques are presented to remove some of the bugs and limitations of Maple. Similar techniques may be applied to other CAS.

First, a technique is presented to constrain the problem of the intermediate expression swell, with an application to a fluid dynamics problem. Second, and the more important is the development of a package **dsolve95**, to solve linear nonhomogeneous ordinary differential equations. The code for **dsolve/series** in Maple V Release 3 was written by an undergraduate student using a naive method proposed in standard text books. **dsolve95** provides the correct answer for many differential equations for which **dsolve/series** gives the wrong answer. In addition, **dsolve95** determines series solutions much more efficiently.

Chapter 2

The calculation of the low-Reynolds-number mobility functions for two unequal spheres

Mathematics–the unshaken Foundation of Sciences, and the plentiful Fountain of Advantage to human affairs.

Isaac Barrow

2.1 Introduction

The low-Reynolds-number mobility functions for two spheres are defined in Kim & Karrila (1991) and are used in many areas of microhydrodynamics (Durlofsky *et al* 1987; Durlofsky & Brady 1989). This work is a continuation of a program started in Jeffrey and Onishi (1984) (also extended in Jeffrey 1992) which aims to calculate all of the mobility functions in a uniform manner. Each function is expressed as a series expansion in the centre-to-centre separation of the spheres, using the method of twin multipole expansions. The expressions are accurate by themselves for large separations, and can be combined with asymptotic analyses available in the literature (Corless & Jeffrey 1988; Jeffrey & Corless 1988; Jeffrey

1989a; Jeffrey 1989b; Jeffrey 1991) to obtain expressions valid for all separations of the spheres. The aim is not simply to print tables of the functions; that would be inefficient, in view of the ease with which data can be distributed on computer disks. Rather, sufficient information is given here to allow numerical evaluation of the functions to a reasonable accuracy, and if greater accuracy is required, the necessary data can be made available. Many of the equations in Jeffrey and Onishi (1984) and Jeffrey (1992) are reused, and it would be wasteful to repeat them here, so instead we shall reference them using the convention that equation numbers prefixed with JO and J refer to equations in these papers.

We start by extending the mobility matrix defined in (JO 1.10). We label two spheres 1 and 2. Sphere α has radius a_α and is centred at $\boldsymbol{x}_\alpha$. It has velocity $\boldsymbol{U}_\alpha$, and angular velocity $\boldsymbol{\Omega}_\alpha$. Both spheres are in an ambient velocity field

$$\boldsymbol{U}(\boldsymbol{x}) = \boldsymbol{U}_\infty + \boldsymbol{\Omega}_\infty \times \boldsymbol{x} + \mathsf{E}_\infty \cdot \boldsymbol{x} \,,$$

where $\boldsymbol{U}_\infty$, $\boldsymbol{\Omega}_\infty$ and E_∞ are constants. The quantities we wish to calculate are the stresslets

$$\mathsf{S}_\alpha = -\int [\tfrac{1}{2}(\boldsymbol{x}'\,\boldsymbol{\sigma}\cdot\boldsymbol{n} + \boldsymbol{\sigma}\cdot\boldsymbol{n}\,\boldsymbol{x}') - \tfrac{1}{3}\mathsf{I}\,\boldsymbol{x}'\cdot\boldsymbol{\sigma}\cdot\boldsymbol{n}]\,dA \,, \tag{2.1}$$

where $\boldsymbol{x}'_\alpha = \boldsymbol{x} - \boldsymbol{x}_\alpha$, a vector from the sphere's centre, $\boldsymbol{\sigma}$ is the stress tensor, $\boldsymbol{n}$ is the unit outward normal, and the integral is over the surface of the sphere. Because of the linearity of the Stokes equations, the above quantities are related to each other through a generalized matrix equation based on straightforward conventions (Kim

and Karrila 1991; section 5.3).

$$\begin{pmatrix} \boldsymbol{U}_1 - \boldsymbol{U}(\boldsymbol{x}_1) \\ \boldsymbol{U}_2 - \boldsymbol{U}(\boldsymbol{x}_2) \\ \boldsymbol{\Omega}_1 - \boldsymbol{\Omega}_\infty \\ \boldsymbol{\Omega}_2 - \boldsymbol{\Omega}_\infty \\ \mu^{-1}\mathbf{S}_1 \\ \mu^{-1}\mathbf{S}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{a}_{11} & \mathbf{a}_{12} & \tilde{\mathbf{b}}_{11} & \tilde{\mathbf{b}}_{12} & \tilde{\mathbf{j}}_1 \\ \mathbf{a}_{21} & \mathbf{a}_{22} & \tilde{\mathbf{b}}_{21} & \tilde{\mathbf{b}}_{22} & \tilde{\mathbf{j}}_2 \\ \mathbf{b}_{11} & \mathbf{b}_{12} & \mathbf{c}_{11} & \mathbf{c}_{12} & \tilde{\mathbf{k}}_1 \\ \mathbf{b}_{21} & \mathbf{b}_{22} & \mathbf{c}_{21} & \mathbf{c}_{22} & \tilde{\mathbf{k}}_2 \\ \mathbf{j}_{11} & \mathbf{j}_{12} & \mathbf{k}_{11} & \mathbf{k}_{12} & \mathbf{n}_1 \\ \mathbf{j}_{21} & \mathbf{j}_{22} & \mathbf{k}_{21} & \mathbf{k}_{22} & \mathbf{n}_2 \end{pmatrix} \begin{pmatrix} \mu^{-1}\boldsymbol{F}_1 \\ \mu^{-1}\boldsymbol{F}_2 \\ \mu^{-1}\boldsymbol{L}_1 \\ \mu^{-1}\boldsymbol{L}_2 \\ \mathbf{E}_\infty \end{pmatrix} . \tag{2.2}$$

The square matrix is the mobility matrix. The elements **a**, **b** and **c** are second-rank tensors, **j** and **k** are third-rank tensors and **n** is fourth rank. It should also be noticed that the physical dimensions of the vectors and matrices in (2.2) are not all the same. The elements **a**, **b** and **c** have been tabulated for unequal spheres (Jeffrey & Onishi 1984) and all of the elements have been calculated for *equal* spheres (Kim & Mifflin 1985). Here we calculate them all for the unequal case. The elements of the matrix obey a number of symmetry conditions, mostly straightforward additions to those given in Jeffrey & Onishi (1984). They can be expressed most easily by adopting suffix notation, in which any tensor $\mathbf{P}_{\alpha\beta}$ is written $P_{ijk}^{(\alpha\beta)}$. We have the following symmetric conditions.

$$\tilde{j}_{ijk}^{\alpha} = j_{jki}^{1\alpha} + j_{jki}^{2\alpha}, \tag{2.3}$$

$$\tilde{k}_{ijk}^{\alpha} = k_{jki}^{1\alpha} + k_{jki}^{2\alpha}, \tag{2.4}$$

$$n_{ijkl}^{1} + n_{ijkl}^{2} = n_{klij}^{1} + n_{klij}^{2}, \tag{2.5}$$

Nondimensionalization, denoted using a circumflex, is effected by setting

$$\mathbf{j}_{\alpha\beta} = (a_\alpha + a_\beta)\hat{\mathbf{j}}_{\alpha\beta} , \tag{2.6}$$

$$\mathbf{k}_{\alpha\beta} = \hat{\mathbf{k}}_{\alpha\beta} , \tag{2.7}$$

$$\mathbf{n}_{\alpha\beta} = \tfrac{5}{6}\pi(a_\alpha + a_\beta)^3\hat{\mathbf{n}}_{\alpha\beta} . \tag{2.8}$$

This scheme has been widely used and leads to tidy symmetry conditions (2.9), but also causes many factors $(1 + \lambda)$ to appear in other equations. Another scheme was

suggested by Kim and Karrila (1991), who replaced the factor $(a_\alpha + a_\beta)$ everywhere in (2.6 to 2.8) with $2a_\alpha$. The first scheme is followed here for consistency with earlier work.

When we express the tensor mobility functions in terms of scalar functions, we can arrange things in such a way that each scalar function is associated with a simple type of flow. As a consequence, a notational scheme can be devised that makes it easy to identify the type of interaction described by a given function. The following scheme was started by Jeffrey and Onishi (1984). The two-sphere geometry is defined by a unit vector along the line of centres,

$$\boldsymbol{d} = (\boldsymbol{x}_2 - \boldsymbol{x}_1)/|\boldsymbol{x}_2 - \boldsymbol{x}_1| \ ,$$

and by the three scalars $r = |\boldsymbol{x}_2 - \boldsymbol{x}_1|$, a_1 and a_2. If ϕ is the azimuthal angle with respect to $\boldsymbol{d}$, then all basic flows are proportional to $\exp(im\phi)$. The letter x is used for any function associated with flows axisymmetric about the line of centres (i.e. $m = 0$), y is used for any function associated with flows transverse to the line of centres ($m = 1$), and z for flows perpendicular to the line of centres ($m = 2$).

Since the tensor functions $\mathbf{j}$, $\mathbf{k}$, $\mathbf{n}$ can contain only the vector $\boldsymbol{d}$, the isotropic tensor $\mathbf{I}$ and scalar quantities, they can be expressed in terms of non-dimensional scalar mobility functions (Kim and Karrila 1991) as follows.

$$\begin{aligned}
\hat{j}^{\alpha\beta}_{ijk} &= x^j_{\alpha\beta}(d_i d_j - \tfrac{1}{3}\delta_{ij})d_k + y^j_{\alpha\beta}(d_i\delta_{jk} + d_j\delta_{ik} - 2d_i d_j d_k), \\
\hat{k}^{\alpha\beta}_{ijk} &= y^k_{\alpha\beta}(d_i\epsilon_{jkl}d_l + d_j\epsilon_{ikl}d_l), \\
\hat{n}^{\alpha\beta}_{ijkl} &= \tfrac{3}{2}x^n_{\alpha\beta}(d_i d_j - \tfrac{1}{3}\delta_{ij})(d_k d_l - \tfrac{1}{3}\delta_{kl}) + \tfrac{1}{2}y^n_{\alpha\beta}(d_i\delta_{jl}d_k + d_j\delta_{il}d_k + d_i\delta_{jk}d_l \\
&\qquad + d_j\delta_{ik}d_l - 4d_i d_j d_k d_l) + \tfrac{1}{2}z^n_{\alpha\beta}(\delta_{ik}\delta_{jl} + \delta_{jk}\delta_{il} - \delta_{ij}\delta_{kl}
\end{aligned}$$

$$+d_id_j\delta_{kl}+\delta_{ij}d_kd_l-d_i\delta_{jl}d_k-d_j\delta_{il}d_k-d_i\delta_{jk}d_l$$
$$-d_j\delta_{ik}d_l+d_id_jd_kd_l)$$

The scalar resistance functions have two non-dimensional arguments made from r, a_1 and a_2. We choose

$$s=2r/(a_1+a_2)\ \text{ and }\ \lambda=a_2/a_1\ .$$

We have the following symmetric conditions.

$$\begin{aligned}
x^a_{\alpha\beta}(s,\lambda) &= x^a_{(3-\alpha)(3-\beta)}(s,\lambda^{-1})\ ,\\
y^a_{\alpha\beta}(s,\lambda) &= y^a_{(3-\alpha)(3-\beta)}(s,\lambda^{-1})\ ,\\
y^b_{\alpha\beta}(s,\lambda) &= -y^b_{(3-\alpha)(3-\beta)}(s,\lambda^{-1})\ ,\\
x^c_{\alpha\beta}(s,\lambda) &= x^c_{(3-\alpha)(3-\beta)}(s,\lambda^{-1})\ ,\\
y^c_{\alpha\beta}(s,\lambda) &= y^c_{(3-\alpha)(3-\beta)}(s,\lambda^{-1})\ ,\\
x^j_{\alpha\beta}(s,\lambda) &= -x^j_{(3-\alpha)(3-\beta)}(s,\lambda^{-1})\ ,\\
y^j_{\alpha\beta}(s,\lambda) &= -y^j_{(3-\alpha)(3-\beta)}(s,\lambda^{-1})\ ,\\
y^k_{\alpha\beta}(s,\lambda) &= y^k_{(3-\alpha)(3-\beta)}(s,\lambda^{-1})\ .
\end{aligned}\qquad (2.9)$$

2.2 Relation to resistance functions

It is important for checking results and deriving asymptotic formulae to know the relations between the mobility functions and the resistance functions defined in Jeffrey & Onishi (1984) and Jeffrey (1992). The nondimensionalized functions obey the following relations.

$$\begin{pmatrix} x^j_{11} & \frac{1+\lambda}{2}x^j_{12}\\ \frac{1+\lambda}{2}x^j_{21} & \lambda\, x^j_{22}\end{pmatrix}=\begin{pmatrix} X^G_{11} & \frac{(1+\lambda)^2}{4}X^G_{12}\\ \frac{(1+\lambda)^2}{4}X^G_{21} & \lambda^2 X^G_{22}\end{pmatrix}\begin{pmatrix} \frac{1}{3}x^a_{11} & \frac{2}{3}\frac{1}{1+\lambda}x^a_{12}\\ \frac{2}{3}\frac{1}{1+\lambda}x^a_{12} & \frac{1}{3\lambda}x^a_{22}\end{pmatrix}\ ,$$

$$\begin{pmatrix} y^j_{11} & \frac{1+\lambda}{2}\, y^j_{12} \\ \frac{1+\lambda}{2}\, y^j_{21} & \lambda\, y^j_{22} \end{pmatrix} = \begin{pmatrix} Y^G_{11} & \frac{(1+\lambda)^2}{4}\, Y^G_{12} \\ \frac{(1+\lambda)^2}{4}\, Y^G_{21} & \lambda^2\, Y^G_{22} \end{pmatrix} \begin{pmatrix} \frac{1}{3}\, y^a_{11} & \frac{2}{3}\,\frac{1}{1+\lambda}\, y^a_{12} \\ \frac{2}{3}\,\frac{1}{1+\lambda}\, y^a_{12} & \frac{1}{3\lambda}\, y^a_{22} \end{pmatrix}$$
$$- \begin{pmatrix} Y^H_{11} & \frac{(1+\lambda)^3}{8}\, Y^H_{12} \\ \frac{(1+\lambda)^3}{8}\, Y^H_{21} & \lambda^3\, Y^H_{22} \end{pmatrix} \begin{pmatrix} y^b_{11} & \frac{4}{(1+\lambda)^2}\, y^b_{12} \\ \frac{4}{(1+\lambda)^2}\, y^b_{21} & \frac{1}{\lambda^2}\, y^b_{22} \end{pmatrix} ,$$

$$\begin{pmatrix} -y^k_{11} & -y^k_{12} \\ -y^k_{21} & -y^j_{22} \end{pmatrix} = \begin{pmatrix} Y^G_{11} & \frac{(1+\lambda)^2}{4}\, Y^G_{12} \\ \frac{(1+\lambda)^2}{4}\, Y^G_{21} & \lambda^2\, Y^G_{22} \end{pmatrix} \begin{pmatrix} y^b_{11} & \frac{4}{(1+\lambda)^2}\, y^b_{21} \\ \frac{4}{(1+\lambda)^2}\, y^b_{12} & \frac{1}{\lambda^2}\, y^b_{22} \end{pmatrix}$$
$$- \begin{pmatrix} Y^H_{11} & \frac{(1+\lambda)^3}{8}\, Y^H_{12} \\ \frac{(1+\lambda)^3}{8}\, Y^H_{21} & \lambda^3\, Y^H_{22} \end{pmatrix} \begin{pmatrix} y^c_{11} & \frac{8}{(1+\lambda)^3}\, y^c_{12} \\ \frac{8}{(1+\lambda)^3}\, y^c_{21} & \frac{1}{\lambda^3}\, y^c_{22} \end{pmatrix} ,$$

$$x^n_1 = X^M_{11} + \frac{1}{8}(1+\lambda)^3\, X^M_{12} - \frac{4}{5}\, X^G_{11}\left[x^j_{11} + \frac{1}{2}(1+\lambda)\, x^j_{21}\right]$$
$$-\frac{1}{10}(1+\lambda)^3\, X^G_{12}\left[x^j_{12} + \frac{2\lambda}{1+\lambda}\, x^j_{22}\right] , \qquad (2.10)$$

$$y^n_1 = Y^M_{11} + \frac{1}{8}(1+\lambda)^3\, Y^M_{12} - \frac{12}{5}\, Y^G_{11}\left[y^j_{11} + \frac{1}{2}(1+\lambda)\, y^j_{21}\right] - \frac{12}{5}\, Y^H_{11}\left[y^k_{11} + y^k_{21}\right]$$
$$-\frac{3}{10}(1+\lambda)^3\, Y^G_{12}\left[y^j_{12} + \frac{2\lambda}{1+\lambda}\, y^j_{22}\right] - \frac{3}{10}(1+\lambda)^3\, Y^H_{12}\left[y^k_{12} + y^k_{22}\right] , \qquad (2.11)$$

$$z^n_1 = Z^M_{11} + \frac{1}{8}(1+\lambda)^3\, Z^M_{12} . \qquad (2.12)$$

2.3 General results used in the calculations

The Cartesian coordinate system we use has its origin at the centre of sphere 1 and a z-axis directed so that the centre of sphere 2 is on the positive axis. The method of twin multipole expansions uses two polar coordinate systems $(\rho_\alpha, \theta_\alpha, \phi)$, one at the centre of each sphere, with the z-axis being a common pole. Different ways of defining these systems were tried in Jeffrey and Onishi (1984) and Happel and Brenner (1973) (their figure 6-3.1). The choice of Jeffrey and Onishi (1984) is

convenient for Laplace's equation, but when used with the Stokes equations it leads to vector products in a left-handed set of axes, which in the long run are a nuisance. This work, however, perseveres with their coordinate system, so that their equations can be reused. The method of twin multipole expansions is described in Jeffrey and Onishi (1984) in sufficient generality to apply to the problems defined here, and so, as stated earlier, equations will not be repeated when they can be referred to there. The only new result needed in order to apply twin-multipole expansions to the problems defined below is an expression for the stresslet of sphere 1. In terms of the pressure coefficients $p^{(1)}_{mn}$ defined in (JO 2.3b), we have

$$\mathbf{S}_1 = 2\pi\mu a_1^2\{p^{(1)}_{02}[\boldsymbol{kk} - \tfrac{1}{3}\mathbf{1}] - p^{(1)}_{12}[\boldsymbol{ik} + \boldsymbol{ki} + \mathrm{i}(\boldsymbol{jk} + \boldsymbol{kj})] \\ +2p^{(1)}_{22}[\boldsymbol{ii} - \boldsymbol{jj} + \mathrm{i}(\boldsymbol{ij} + \boldsymbol{ji})]\} \ . \tag{2.13}$$

2.4 The Mobility functions $x^j_{\alpha\beta}$

These functions describe the stresslets resulting from constant forces parallel to the line of centres, or equivalently the velocities resulting from axisymmetric pure-straining motion. In particular, if the external forces acting on the spheres are $F_\alpha \boldsymbol{k}$, then the stresslet of sphere 1 is given by

$$\mathbf{S}_1 = [2a_1 x^j_{11} F_1 + (a_1 + a_2) x^j_{12} F_2](\boldsymbol{kk} - \tfrac{1}{3}\mathbf{1}) \ . \tag{2.14}$$

We follow Jeffrey and Onishi (1984) and convert this equation into one using equivalent velocity U by

$$(6\pi\mu a_1)^{-1} F_1 \ = \ -(6\pi\mu a_2)^{-1} F_2 = U\boldsymbol{k} \ .$$

Then using (2.13) we obtain

$$x_{11}^{J}U - \frac{1}{2}(1+\lambda)\lambda x_{12}^{J}U \;=\; p_{02}^{(1)}/6\ .$$

The calculation of $p_{02}^{(1)}$ is undertaken in the next section.

Twin multipole expansions

The general series expressions can be obtained from the problems studied in JO section 8. In terms of the quantity P_{npq} defined in (JO 8.17) we have

$$x_{11}^{J} - \tfrac{1}{2}(1+\lambda)\lambda x_{12}^{J} = \tfrac{1}{4}\sum_{p=0}^{\infty}\sum_{q=0}^{\infty} P_{2pq}t_1^p t_2^q\ , \tag{2.15}$$

where $t_\alpha = a_\alpha/r$. For the complementary problem (JO 3.11), we obtain

$$x_{11}^{J} + \tfrac{1}{2}(1+\lambda)\lambda x_{12}^{J} = \tfrac{1}{4}\sum_{p=0}^{\infty}\sum_{q=0}^{\infty}(-1)^{p+q+3}P_{2pq}t_1^p t_2^q\ . \tag{2.16}$$

We conclude that x_{11}^{J} will be a sum over terms for which $p+q$ is odd, while x_{12}^{J} will be a sum with $p+q$ even. Thus

$$x_{11}^{J}(s,\lambda) = \sum_{\substack{m=1\\ m\ \text{odd}}}^{\infty}\sum_{q=0}^{m}\tfrac{1}{4}P_{2(m-q)q}\lambda^q\frac{2^m}{(1+\lambda)^m s^m} = \sum_{\substack{m=1\\ m\ \text{odd}}}^{\infty}\frac{f_m(\lambda)}{(1+\lambda)^m s^m}\ , \tag{2.17}$$

and

$$\begin{aligned} x_{12}^{J}(s,\lambda) \;&=\; \frac{-2}{(1+\lambda)\lambda}\sum_{\substack{m=2\\ m\ \text{even}}}^{\infty}\sum_{q=0}^{m}\tfrac{1}{4}P_{2(m-q)q}\lambda^q\frac{2^m}{(1+\lambda)^m s^m} \\ &= \frac{-2}{(1+\lambda)\lambda}\sum_{\substack{m=2\\ m\ \text{even}}}^{\infty}\frac{f_m(\lambda)}{(1+\lambda)^m s^m}\ . \end{aligned} \tag{2.18}$$

We now give the first 11 terms explicitly.

$$f_0 = f_1 = 0\ ,\quad f_2 = 5\lambda\ ,\quad f_3 = 0\ ,$$

$$f_4 = -12\lambda - 20\lambda^3\ ,\quad f_5 = 200\lambda^3\ ,$$

$$f_6 = 0 \ , \quad f_7 = -1760\lambda^3 + 640\lambda^5 \ ,$$

$$f_8 = 8000\lambda^4 \ , f_9 = 3840\lambda^3 - 20736\lambda^5 + 3840\lambda^7 \ ,$$

$$f_{10} = -12800\lambda^4 + 16000\lambda^6 \ ,$$

$$f_{11} = 89600\lambda^5 + 320000\lambda^6 - 189440\lambda^7 + 19200\lambda^9 \ .$$

2.5 The Mobility functions $y^j_{\alpha\beta}$

These functions describe the stresslets of two spheres due to constant forces transverse to their line of centres, or equivalently they describe the velocities on two spheres in a transverse pure-straining motion. If the external forces acting on the sphere α is $F_\alpha \boldsymbol{i}$, then the stresslet is

$$\mathbf{S}_1 = [2a_1 y^j_{11} U_1 + (a_1 + a_2) y^j_{12} U_2](\boldsymbol{ki} + \boldsymbol{ik}) \ . \tag{2.19}$$

Again we convert to an equivalent velocity U_α in order to extract the solution from the calculations performed in Jeffrey and Onishi (1984). See (JO 9.1).

Twin multipole expansions

These functions can be obtained from the problems studied in JO section 9. In terms of the quantity P_{npq} defined in (JO 9.14) we have

$$y^j_{11} - \tfrac{1}{2}(1+\lambda)\lambda y^j_{12} = \tfrac{1}{4} \sum_{p=0}^{\infty} \sum_{q=0}^{\infty} P_{2pq} t_1^p t_2^q \ . \tag{2.20}$$

The complementary problem is parallel to (2.16) and so we conclude that y^j_{11} consists of terms for which $p+q$ is odd and y^j_{12} of terms for which $p+q$ is even.

$$y^j_{11}(s,\lambda) \ = \ \sum_{\substack{m=1 \\ m \text{ odd}}}^{\infty} \sum_{q=0}^{m} \tfrac{1}{4} P_{2(m-q)q} \lambda^q \frac{2^m}{(1+\lambda)^m s^m} = \sum_{\substack{m=1 \\ m \text{ odd}}}^{\infty} \frac{f_m(\lambda)}{(1+\lambda)^m s^m} \ , \tag{2.21}$$

and

$$y_{12}^{j}(s,\lambda) = \frac{-2}{(1+\lambda)\lambda} \sum_{\substack{m=2 \\ m \text{ even}}}^{\infty} \sum_{q=0}^{m} \tfrac{1}{4} P_{2(m-q)q} \lambda^q \frac{2^m}{(1+\lambda)^m s^m} \qquad (2.22)$$

$$= \frac{-2}{(1+\lambda)\lambda} \sum_{\substack{m=2 \\ m \text{ even}}}^{\infty} \frac{f_m(\lambda)}{(1+\lambda)^m s^m} .$$

where

$$f_0 = f_1 = f_2 = f_3 = 0 ,$$

$$f_4 = 4\lambda + \frac{20}{3}\lambda^3 , f_5 = 0 ,$$

$$f_6 = 0 , f_7 = -\frac{400}{3}\lambda^3 + \frac{560}{3}\lambda^5 ,$$

$$f_8 = 0 , f_9 = \frac{2560}{3}\lambda^3 - \frac{8224}{3}\lambda^5 + 1120\lambda^7 ,$$

$$f_{10} = -\frac{11200}{3}\lambda^4 + \frac{8000}{3}\lambda^6 ,$$

$$f_{11} = 22400\lambda^5 - 31232\lambda^7 + 5760\lambda^9 .$$

2.6 The Mobility functions $y_{\alpha\beta}^{k}$

These functions describe the stresslets of two spheres due to couples acting transverse to their line of centres, or equivalently they describe the angular velocities of two spheres in a transverse straining motion. In particular, we let the couples be $L_\alpha \boldsymbol{j}$, so that the stresslet becomes

$$\mathbf{S}_1 = (y_{11}^{k} L_1 + y_{12}^{k} L_2)(\boldsymbol{ik} + \boldsymbol{ki}) .$$

In order to use the calculations of JO section 11, we convert the applied couples to equivalent angular velocities by the equations $L_\alpha = 8\pi\mu a_\alpha^3 \Omega_\alpha \boldsymbol{j}$. We select the Ω_α so that the spheres have the same effective surface velocity, namely, $a_1\Omega_1 = a_2\Omega_2 = U$. With these substitutions, we can re-use the solutions obtained in JO section 9.

Twin multipole expansions

These functions can be obtained from the problems studied in JO section 9. In terms of the quantity P_{npq} defined in (JO 9.14) we have

$$y_{11}^k + \lambda^2 y_{12}^k = -\tfrac{3}{8} \sum_{p=0}^{\infty} \sum_{q=0}^{\infty} P_{2pq} t_1^p t_2^q \,. \tag{2.23}$$

The complementary problem is parallel to (2.16) and so we conclude that y_{11}^k consists of terms for which $p+q$ is even and y_{12}^k of terms for which $p+q$ is odd.

$$y_{11}^k(s,\lambda) = \sum_{\substack{m=0 \\ m \text{ even}}}^{\infty} \sum_{q=0}^{m} -\tfrac{3}{8} P_{2(m-q)q} \lambda^q \frac{2^m}{(1+\lambda)^m s^m} = \sum_{\substack{m=0 \\ m \text{ even}}}^{\infty} \frac{f_m(\lambda)}{(1+\lambda)^m s^m} \,, \tag{2.24}$$

and

$$y_{12}^k(s,\lambda) = \frac{1}{\lambda^2} \sum_{\substack{m=1 \\ m \text{ odd}}}^{\infty} \sum_{q=0}^{m} -\tfrac{3}{8} P_{2(m-q)q} \lambda^q \frac{2^m}{(1+\lambda)^m s^m} = \frac{1}{\lambda^2} \sum_{\substack{m=1 \\ m \text{ odd}}}^{\infty} \frac{f_m(\lambda)}{(1+\lambda)^m s^m} \,. \tag{2.25}$$

where

$$f_0 = f_1 = f_2 = 0,\ f_3 = 10\lambda^2,\ f_4 = f_5 = 0 \,,$$

$$f_6 = -200\lambda^3 \,, f_7 = 0,\ f_8 = 1280\lambda^3 - 1280\lambda^4 \,,$$

$$f_9 = 40000\lambda^5 \,, f_{10} = 22400\lambda^5 - 9600\lambda^7 \,, f_{11} = 64000\lambda^7 \,.$$

2.7 The Mobility functions x_α^n, y_α^n, z_α^n

Using the relation (2.10) we obtain

$$x_1^n = \sum_{m=0}^{\infty} \frac{f_m(\lambda)}{(1+\lambda)^m s^m} \,,$$

where

$$f_0 = f_1 = f_2 = 0,\ f_3 = 40\lambda^3,\ f_4 = 0, f_5 = -192\lambda^3 - 192\lambda^5,$$

$$f_6 = 1600\lambda^3 \,, f_7 = 0,\ f_8 = 1920\lambda^3(-8 + 5\lambda^2) \,, f_9 = 6400\lambda^6 \,,$$

$$f_{10} = 3072\lambda^3(12 - 81\lambda^2 + 25\lambda^4) \,, f_{11} = 76800\lambda^6(1 + \lambda^2) \,.$$

Relation (2.11) gives

$$y_1^n = \sum_{m=0}^{\infty} \frac{f_m(\lambda)}{(1+\lambda)^m s^m} ,$$

where

$$f_0 = 1,\ f_1 = f_2 = 0,\ f_3 = -20\lambda^3,\ f_4 = 0, f_5 = 128\lambda^3(1+\lambda^2),$$

$$f_6 = 400\lambda^3\ , f_7 = 0,\ f_8 = 1280\lambda^3(-4+5\lambda^2)\ , f_9 = -8000\lambda^6\ ,$$

$$f_{10} = 2048\lambda^3(8-54\lambda^2+25\lambda^4)\ , f_{11} = 76800\lambda^6(1+\lambda^2)\ .$$

From the relation (2.12) we obtain the mobility function as

$$z_1^n = \sum_{m=0}^{\infty} \frac{f_m(\lambda)}{(1+\lambda)^m s^m} ,$$

where

$$f_0 = 1,\ f_1 = f_2 = f_3 = f_4 = 0,$$

$$f_5 = -32\lambda^3 - 32\lambda^5, f_6 = f_7 = 0, f_8 = 800\lambda^5,$$

$$f_9 = 0\ , f_{10} = 1024\lambda^3 - 6912\lambda^5 + 8960\lambda^7\ , f_{11} = 0\ .$$

To confirm the correctness of the calculations, the inverse relations with the resistance functions were checked using Maple. The separately computed series for the various functions were substituted into the equations and the two sides compared to $O(s^{-11})$.

Chapter 3

Inviscid irrotational flow past a sphere touching a plane.

Mathematics, rightly viewed, possesses not only truth, but supreme beauty–a beauty cold and austere, like that of sculpture.

Bertrand Russell, *Mysticism and Logic, 1918.*

Latta & Hess (1973) studied the problem of incompressible potential flow past a sphere in contact with a plane. The velocity at large distance is taken to be uniform and parallel to the plane. This problem is a simple example where we can exploit Maple in order to solve a second order ordinary differential equation with a singular point at the origin. Because of the singular point in the differential equation, it is usually solved numerically by using a series expansion about the singular point to get the first steps of the numerical solution. Here we shall replace the numerical solution with a large number of terms in the series solution. By expanding this series solution along the axis and matching it to an asymptotic solution, we obtain a solution that is more accurate than a numerical one, with less effort.

3.1 Description of the problem

Inviscid irrotational flow obeys $\nabla^2\Phi = 0$, where Φ is the velocity potential, defined by (Batchelor 1967)

$$\underline{u} \;=\; \nabla\Phi.$$

The boundary condition for Φ is that $\underline{n}.\nabla\Phi = \underline{V}.\underline{n}$, where $\underline{V}$ is the velocity of the boundary. In our case, the boundaries are not moving, so

$$\underline{n}.\nabla\Phi = \frac{\partial\Phi}{\partial n} \;=\; 0.$$

At large distances from the sphere, the flow approaches that of a uniform stream, so we write, in cylindrical coordinates (z, θ, ρ)

$$\Phi \;=\; \rho\cos\theta + \cos\theta\,\phi(z,\rho),$$

where the dependence on θ has been removed from ϕ. Using tangent-sphere coordinates (ξ, η, θ) (Moon & Spencer 1961), defined in terms of cylindrical polar coordinates by

$$z = \frac{2\xi}{\xi^2+\eta^2} \qquad \rho = \frac{2\eta}{\xi^2+\eta^2},$$

ϕ can be written as an integral transform

$$\phi \;=\; \int_0^\infty F(s)\cosh s\xi\, J_1(s\eta)\,ds, \tag{3.1}$$

subject to the boundary condition

$$\frac{\partial\phi}{\partial\xi} \;=\; 0 \qquad \text{at} \quad \xi = 0, 1.$$

The equation for $F(s)$ is transformed to a second order ordinary differential equation

$$s \sinh s\, G''(s) + 3 \sinh s\, G'(s) - (s \sinh s + \cosh s) G(s) \quad = \quad -2 \sinh s\, e^{-s} \quad (3.2)$$

where

$$G(s) \quad = \quad s^{-1} \sinh s\, F(s),$$

subject to the boundary conditions that the integral (3.1) be convergent.

At $s = 0$, the integrand of (3.1), $F(s) \cosh s\xi\, J_1(s\eta)$ is asymptotic to $F(s)\, s\, \eta$.

Let $F(s) \sim s^{\alpha}$, then the integral is convergent at $s = 0$, if $\alpha > -2$.

At $s = \infty$, $F(s) \cosh s\xi\, J_1(s\eta)$ is asymptotic to $F(s)\, e^{s\xi}\, (s\eta)^{-\frac{1}{2}} \cos s\eta, \quad \xi < 1$.

Let $F(s) \sim e^{-\beta s}$, then the integral (3.1) is convergent at $s = \infty$, if $\beta \geq 1$.

To find the series solutions, we use the new routines developed in chapter 6 which enable us to obtain a large number of terms in the solutions easily and quickly. Moreover, we re-expand the solution at different points along the real axis thus working around the convergence problem and obtain a highly accurate solution as follows.

3.2 Series solutions about the singular point at the origin

The differential equation (3.2) has regular singular points, where $\sinh s = 0$, at the origin and in the complex plane at $s = i\, n\, \pi$. o find the series solution of the differential equation (3.2) about the singular point at origin, the routine `dsolve/series` in Maple can be used, but due to some errors in the existing routine in Maple V Release 3, it provides incorrect homogeneous solutions and no particular integral.

To obtain correct homogeneous solutions and the particular integral about origin, we use the routines developed in the chapter 6. We denote the homogeneous solutions about $s = 0$ as $G_h^{(0)}$ and the particular integral as $G_p^{(0)}$. Using Maple we found series solutions correct to 50 terms.

$$G_h^{(0)} = s^{(-1+\sqrt{2})}[1 + \sum_{\substack{n=2 \\ n\,\text{even}}}^{50} a_n s^n],$$

where first few constants in the series are

$$\begin{aligned} a_2 &= -\frac{1}{3} + \frac{1}{3}\sqrt{2}\,, \\ a_4 &= -\frac{41}{360} + \frac{61}{720}\sqrt{2}\,, \\ a_6 &= -\frac{67}{8820} + \frac{347}{63504}\sqrt{2}\,, \\ \ldots &\quad \ldots \quad \ldots \end{aligned}$$

The particular integral $G_p^{(0)}$, correct to 50 terms is found as

$$G_p^{(0)} = \sum_{n=1}^{50} b_n s^n,$$

where

$$\begin{aligned} b_1 &= -1, \qquad b_2 = \frac{2}{7}\,, \\ b_3 &= -\frac{1}{6}, \qquad b_4 = \frac{5}{161}\,, \\ \ldots &\quad \ldots \quad \ldots \end{aligned}$$

The other homogeneous solution is asymptotically $O(s^{-(\sqrt{2}+1)})$, for small values of s. The convergence of the integral (3.1) at $s = 0$ requires that the constant associated with this homogeneous solution equals zero. Hence for small s, the general solution of differential equation (3.2) is

$$G(s) = G_p^{(0)} + k\,G_h^{(0)},$$

where the constant k has to be determined.

3.3 Series solutions about other expansion points

The solutions of the differential equation (3.2) may be extended along the real axis by analytic or numerical continuation. We use the solutions $G_h^{(0)}$ and $G_p^{(0)}$ to give us boundary conditions at a new expansion point. We begin with the series solutions about $s = 5/2$, an ordinary point of the differential equation (3.2). We find the homogeneous solution $G_h^{(5/2)}$ of the homogeneous differential equation (3.2) [with right hand side set to 0] subject to the boundary conditions

$$G_h^{(5/2)} = G_h^{(0)}\Big|_{s=5/2}, \qquad \frac{d}{ds} G_h^{(5/2)} = \frac{d}{ds} G_h^{(0)}\Big|_{s=5/2}$$

Similarly, we find the particular integral $G_p^{(5/2)}$ of the nonhomogeneous differential equation (3.2) subject to the boundary conditions

$$G_p^{(5/2)} = G_p^{(0)}\Big|_{s=5/2}, \qquad \frac{d}{ds} G_p^{(5/2)} = \frac{d}{ds} G_p^{(0)}\Big|_{s=5/2}.$$

The general solution of the differential equation (3.2) about $s = 5/2$ is

$$G(s) \;=\; G_p^{(5/2)} + k\, G_h^{(5/2)}.$$

We now use these solutions to give us boundary conditions at a new expansion point $s = 4$. We find the homogeneous solution $G_h^{(4)}$ of the homogeneous differential equation subject to the boundary conditions

$$G_h^{(4)} = G_h^{(5/2)}\Big|_{s=4}, \qquad \frac{d}{ds} G_h^{(4)} = \frac{d}{ds} G_h^{(5/2)}\Big|_{s=4},$$

and the particular integral $G_p^{(4)}$ subject to the boundary conditions

$$G_p^{(4)} = G_p^{(5/2)}\Big|_{s=4}, \qquad \frac{d}{ds} G_p^{(4)} = \frac{d}{ds} G_p^{(5/2)}\Big|_{s=4}.$$

The general solution of the differential equation (3.2) about $s = 4$ is

$$G(s) = G_p^{(4)} + k\, G_h^{(4)}.$$

Proceeding in this manner, we find the solutions at expansion points $s = 6$, $s = 8$ etc. All the solutions were found correct to 50 terms to achieve high accuracy. Figures (3.1) and (3.2) show the values of s where the series expansions are accurate.

3.4 Asymptotic solution

About $s = \infty$, we obtain an asymptotic expansion as follows.

As a first order approximation, we approximate $\coth s$ as 1 in (3.2) and find the particular integral g_p and the homogeneous solution g_h for the resulting differential equation as

$$g_h = \frac{1}{s^2}\, e^{-s}, \qquad g_p = \frac{1}{2}\, e^{-s}.$$

We neglect the other homogeneous solution which is $s^{-2}(2\,s - 1)e^{s}$ for large s.

For the next order approximation, $\coth(s) \sim 1 + 2\, e^{-2s}$. We consider the homogeneous solution g_{h2} in the form

$$g_{h2} = \frac{e^{-s}}{s^2} + f(s)\, e^{-3s}.$$

Substitute g_{h2} in the differential equation (3.2), we solve the differential equation for $f(s)$ and get g_{h2} as

$$g_{h2} = \frac{e^{-s}}{s^2} - \frac{e^{-3s}}{s^2} + 2\frac{e^{-s}\mathrm{Ei}(1,2s)}{s^2} - 2\frac{e^{s}\mathrm{Ei}(1,4s)}{s^2} + 4\frac{e^{s}\mathrm{Ei}(1,4s)}{s}.$$

Similarly, we find the particular integral g_{p2} as

$$g_{p2} = \frac{1}{2}\, e^{-s} + \left(\frac{1}{16s^2} + \frac{1}{8s}\right) e^{-3s}.$$

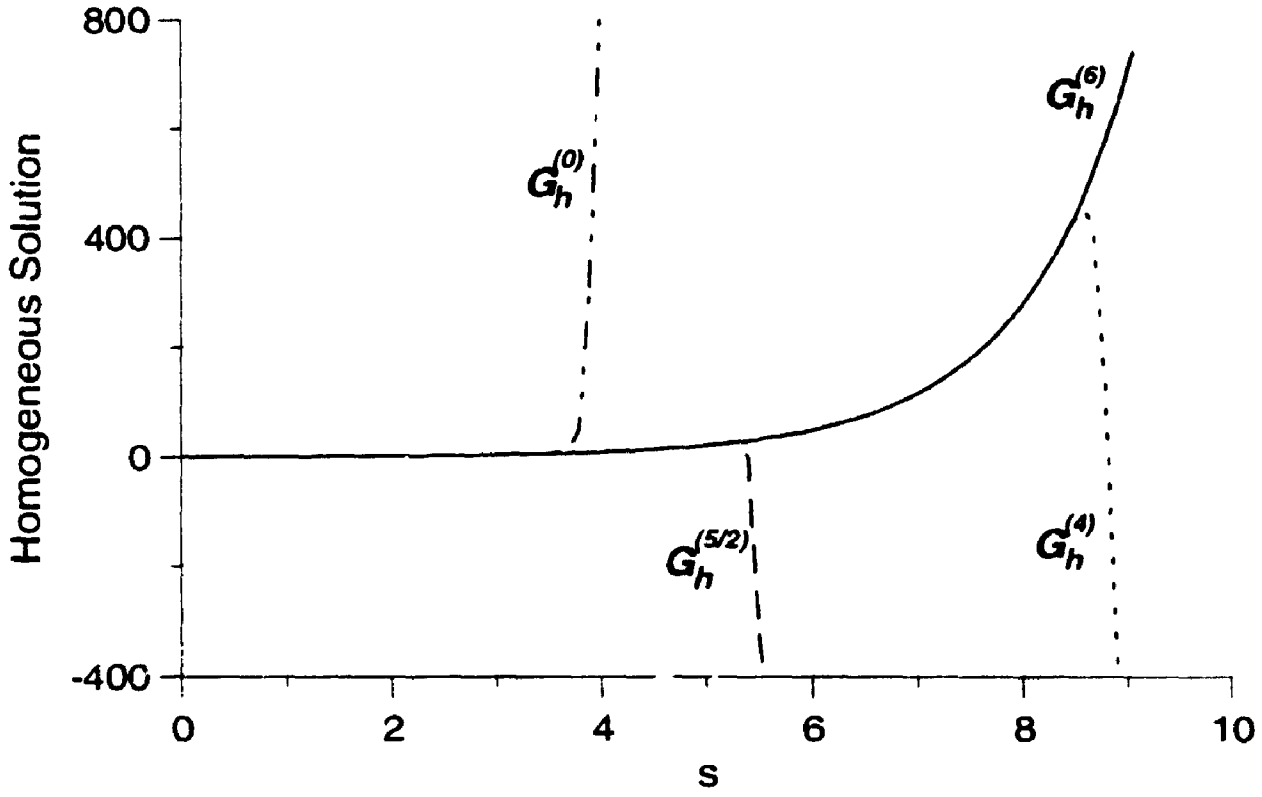

Figure 3.1: Series expansions for the homogeneous solution about $s = 0, 5/2, 4$ and 6.

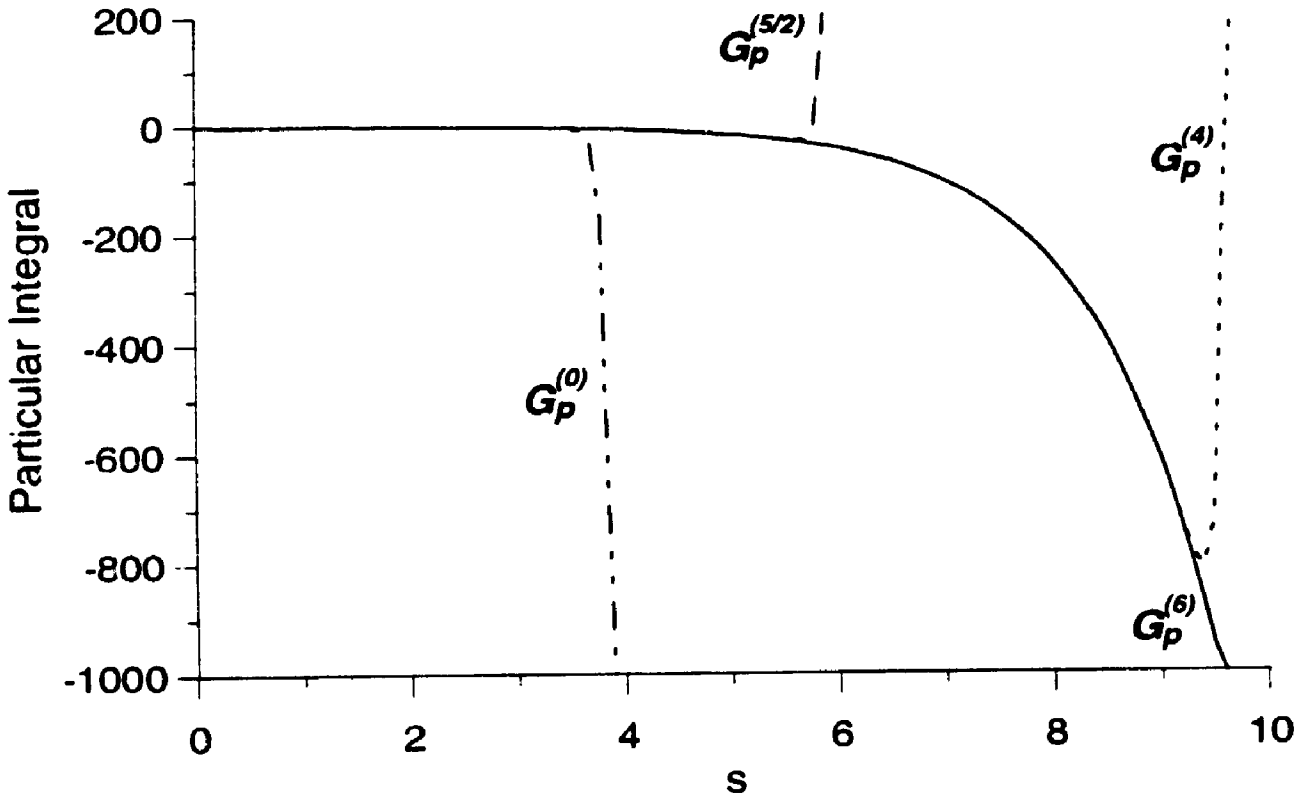

Figure 3.2: Series expansions for the particular integral about $s = 0, 5/2, 4$ and 6.

Hence for large s, differential equation (3.2) has asymptotic solution

$$G(s) \sim g_{p2} + K\, g_{h2}\,, \tag{3.3}$$

where the constant K has to be determined.

Complete Solution. We have therefore obtained the solution of the differential equation (3.2) in the form

$$G(s) = \begin{cases} G_p^{(0)} + k\, G_h^{(0)} & 0 \le s < 2.5\,, \\ G_p^{(5/2)} + k\, G_h^{(5/2)} & 2.5 \le s < 4\,, \\ G_p^{(4)} + k\, G_h^{(4)} & 4 \le s < 6\,, \\ G_p^{(6)} + k\, G_h^{(6)} & 6 \le s < 8\,, \\ \vdots & \vdots \\ g_{p2} + K\, g_{h2} & s \to \infty\,. \end{cases}$$

Our final step is to determine the constants K and k.

3.5 The Constants K and k

We find these constants by matching the two solutions, $G_p^{(6)} + k\, G_h^{(6)}$ and $g_{p2} + K\, g_{h2}$. The two solutions and their derivatives are equal for those values of s where these solutions are accurate. Let the matching point be s_m. Then we require

$$\begin{aligned} G_p^{(6)}(s_m) + k\, G_h^{(6)}(s_m) &= g_{p2}(s_m) + K\, g_{h2}(s_m)\,, \\ \left.\frac{dG_p^{(6)}}{ds}\right|_{s=s_m} + k \left.\frac{dG_h^{(6)}}{ds}\right|_{s=s_m} &= \left.\frac{dg_{p2}}{ds}\right|_{s=s_m} + K \left.\frac{dg_{h2}}{ds}\right|_{s=s_m}. \end{aligned}$$

By choosing different values of s_m in the range $6 \le s \le 8$ we obtain the constants as given in Table (3.1). The programs are given in Appendix A.

s_m	K	k
6.0	-.070325647266	.8955928431577866881692
6.4	-.070325647288	.8955928431577866653188
6.8	-.070325647293	.8955928431577866632507
7.2	-.070325647294	.8955928431577866630635
7.6	-.070325647294	.8955928431577866630466
8.0	-.070325647294	.8955928431577866630450

Table 3.1: Values of constants K and k at various matching points in the range $6 \leq s \leq 8$.

Chapter 4

Motion of a sphere near a wall at low Reynolds number.

Give us the tools, and we will finish the job.

Winston Churchill, *Feb 9, 1941; Addressing President Roosevelt.*

Part I. Lubrication theory using large-expression management in Maple.

4.1 Introduction

Lubrication theory is an example of a perturbation calculation. Traditional engineering lubrication theory is actually the first approximation in a sequence of approximations. A full treatment of lubrication theory requires the apparatus of inner and outer expansions but in many applications, this can be avoided. Lubrication theory divides the flow domain into a gap region and a remaining region outside the gap. Because the singular terms in expressions for lubrication forces come mostly from the gap analysis, the flow outside the gap can be neglected in a first approximation. In the gap region, the equations can be solved as a succession of regular

perturbations. The main difficulty that must be faced in deriving gap solutions to higher order is the existence of expression swell.

The problem of expression swell has been known to mathematicians for generations, and they have developed many strategies for carrying a calculation forward in spite of the repulsive sight of equations growing ever longer as they feed on their predecessors. The literature contains many heroic perturbation solutions to high order (Van Dyke 1974; Delaunay 1867; Deprit, Henrard and Rom 1970) and their existence, and correctness, is a testimony not only to human endurance but also to the efficacy of the strategies that the human calculators used to manage their intermediate expressions.

Since one of the attractions of computer algebra systems is their ability to manipulate long expressions, it is natural to turn to them to continue calculations that stopped because of expression swell. However, simply transferring perturbation calculations to a CAS does not automatically mean that they can be extended to the required order. In particular, the problem studied here leads to expressions that increase in size so rapidly that they exhaust the memory of any present machine. For this reason, we have considered how the successful strategies of the pre-computer age could be adapted to CAS. Very few of the strategies developed by mathematicians to cope with large expressions are reflected in current algebra systems. One suspects that system developers think they are no longer necessary, but such thoughts would be erroneous, as the problem in this chapter shows.

In addition to the straightforward problem of memory, another problem of importance to mathematicians is the comprehension of large expressions. A recent

description of using Maple in the classroom (Boyce and Ecker 1992) commented that students laughed while an unexpectedly large expression scrolled across the screen. The reaction of the class is indicative of the discomfort many users feel with very large expressions. Current systems offer only a little assistance in this area: Mathematica will hide the expression from you, and Maple will attempt to identify subexpressions which can be printed separately, but in neither case is there a way of displaying the mathematical skeleton of an expression. In the past, a persistent human might tackle an equation that spreads over one or two pages, but now some computer users are undaunted by an equation that take 150 pages to print and clearly most people would find it difficult to extract an overview of such an equation.

The key idea developed here is called hierarchical representation. Related ideas have appeared in the literature and in software systems under the names such as computation sequences and straight line programs. These techniques have concentrated on obtaining computation sequences from long expressions **after** they have been derived. For example, the roots of a cubic, say

$$p(x) = x^3 + a\,x + 1,$$

are printed in Maple as

$$\begin{aligned} x &= \%1^{1/3} - \frac{1}{3}\,\frac{a}{\%1^{1/3}} \\ &= -\frac{1}{2}\,\%1^{1/3} + \frac{1}{6}\,\frac{a}{\%1^{1/3}} + \frac{1}{2}\,I\,3^{1/2}\left(\%1^{1/3} + \frac{1}{3}\,\frac{a}{\%1^{1/3}}\right) \\ &= -\frac{1}{2}\,\%1^{1/3} + \frac{1}{6}\,\frac{a}{\%1^{1/3}} - \frac{1}{2}\,I\,3^{1/2}\left(\%1^{1/3} + \frac{1}{3}\,\frac{a}{\%1^{1/3}}\right) \\ \%1 &:= -\frac{1}{2} + \frac{1}{18}(12\,a^3 + 81)^{1/2} \end{aligned}$$

The common subexpression denoted by %1 is automatically identified by Maple's pretty printer and makes the printed solution very similar to the solution formulae given in, for example, Abramowitz & Stegun (1970). The explicit form of the roots, as desired by a naive user, is no more useful and rather more unwieldy. Further, with regard to numerical evaluation, direct application of stable numerical methods to the original polynomial, given a numerical value of a, is usually a better idea.

Our technique differs from those listed above in that it generates a sequence *during* the derivation of long expressions, and further, uses the sequences in subsequent calculations. This approach has an advantage that the sequences are natural to the problem at hand and their forms can be controlled by the user. Since the simplifications can be introduced in the early stages of the calculation, their advantages can be felt and seen through the rest of the calculation in terms of quicker processing and smaller memory requirements.

The problem of lubrication theory for a sphere moving parallel to a plane wall is a special case of the flow between two rigid spheres in the lubrication approximation that has been solved by Jeffrey (1982) through a perturbation scheme. He expressed the solution with carefully chosen intermediate variables. A straight forward implementation of the same scheme in Maple gives a large solution which is very difficult to comprehend. Maple does have a automatic subexpression optimizer but it is not intelligent enough to provide the same compact form as given by Jeffrey (1982). In this chapter we extend the solution to higher order by using hierarchical representation techniques in Maple.

4.2 Problem

Two spheres of different radii are approaching each other with equal and opposite velocities, the fluid around them being at low Reynolds number. For the purpose of calculation, consider a sphere of radius a moving with velocity U towards a second stationary sphere of radius $-a/\kappa$ (so that κ is negative). Choosing this situation allows one to set $\kappa = 0$ and obtain the special case of a sphere moving towards a plane. The minimum separation of the two spheres is εa with $\varepsilon << 1$. Because the spheres are moving along their line of centers, the flow is axisymmetric and one can define a stream function ψ so that, in cylindrical coordinates (r, θ, z), the fluid velocity is expressed in terms of a stream function ψ as

$$u = U(r^{-1}\psi_z, 0, -r^{-1}\psi_r).$$

The Stokes equation for ψ is

$$\nabla^4 \psi = \left(\frac{\partial^2}{\partial z^2} + \frac{\partial^2}{\partial r^2} - \frac{1}{r}\frac{\partial}{\partial r}\right)^2 \psi = 0. \tag{4.1}$$

The boundary conditions are

$$\psi = \frac{1}{2}r^2, \qquad \psi_z = 0\ ,$$

on the moving sphere and

$$\psi = \psi_z = 0\ ,$$

on the stationary sphere. The perturbation solution is based on the observation that when the spheres are separated by a gap h which is small, their surfaces can be approximated as follows. Stretching the coordinates using

$$Z = \frac{z}{a\varepsilon} \qquad \text{and} \qquad R = \frac{r}{a\varepsilon^{1/2}},$$

we choose the point $Z = 0$, $R = 0$ on the surface of the stationary sphere at the narrowest part of the gap, (4.1) then becomes

$$\left[\frac{\partial^2}{\partial Z^2} + \varepsilon \left(\frac{\partial^2}{\partial R^2} - \frac{1}{R}\frac{\partial}{\partial R}\right)\right]^2 \psi = 0. \tag{4.2}$$

The surfaces of the moving and the stationary spheres are given by $(z-a-h)^2+r^2 = a^2$ and $(z - a/\kappa)^2 + r^2 = a^2/\kappa^2$. If the surfaces are given in stretched coordinates by $Z = Z_1(R)$ and $Z = Z_2(R)$, then we can expand the above to get

$$\begin{aligned} Z_1 &= H_1 + \frac{1}{8}\varepsilon R^4 + \frac{1}{16}\varepsilon^2 R^6 + O(\varepsilon^3)\,, \\ Z_2 &= H_2 + \frac{1}{8}\varepsilon \kappa^3 R^4 + \frac{1}{16}\varepsilon^2 \kappa^5 R^6 + O(\varepsilon^3), \end{aligned}$$

where

$$\begin{aligned} H_1 &= 1 + \frac{1}{2}R^2\,, \\ H_2 &= \frac{1}{2}\kappa R^2. \end{aligned}$$

We notice that this expansion defines a hierarchy of approximations to be spheres' surfaces. In our Maple procedures, we shall reflect this by defining quantities H_1 and H_2.

Define the operator Υ

$$\Upsilon = \frac{\partial^2}{\partial R^2} - \frac{1}{R}\frac{\partial}{\partial R},$$

which reduces (4.2) to

$$\left[\frac{\partial^4}{\partial Z^4} + 2\varepsilon\Upsilon\frac{\partial^2}{\partial Z^2} + \varepsilon^2\Upsilon^2\right]\psi = 0. \tag{4.3}$$

Look for a solution for ψ in the form of expansion

$$\psi = a^2\varepsilon\left(\psi^{(0)} + \varepsilon\psi^{(1)} + \varepsilon^2\psi^{(2)}\right) + O(\varepsilon^3) \tag{4.4}$$

and derive equations for the $\psi^{(i)}$. These quantities define a natural hierarchy, the members of which we shall calculate successively. Substituting (4.4) into (4.3) and equating the powers of ε, we obtain

$$\frac{\partial^4}{\partial Z^4}\psi^{(0)} = 0, \tag{4.5}$$

$$\frac{\partial^4}{\partial Z^4}\psi^{(1)} = -2\Upsilon\frac{\partial^2\psi^{(0)}}{\partial Z^2}, \tag{4.6}$$

$$\frac{\partial^4}{\partial Z^4}\psi^{(2)} = -2\Upsilon\frac{\partial^2\psi^{(1)}}{\partial Z^2} - \Upsilon^2\psi^{(0)}. \tag{4.7}$$

4.2.1 Boundary conditions for $\psi^{(i)}$

On the moving sphere, $\psi = \frac{1}{2}R^2a^2\varepsilon$. Hence from (4.4),

$$a^2\varepsilon\left[\psi^{(0)}(Z_1,R) + \varepsilon\,\psi^{(1)}(Z_1,R) + \varepsilon^2\,\psi^{(2)}(Z_1,R)\right] + O(\varepsilon^3) = \frac{1}{2}R^2a^2\varepsilon. \tag{4.8}$$

Since Z_1 is a function of ε, it is convenient to shift the boundary condition to $Z_1 = H_1$. For this we use Taylor series to get

$$\begin{aligned}\psi^{(0)}(Z_1,R) &= \psi^{(0)}\left(H_1 + \frac{\varepsilon R^4}{8} + \frac{\varepsilon^2R^6}{16}, R\right)\\ &= \psi^{(0)}(H_1,R) + \left(\frac{\varepsilon R^4}{8} + \frac{\varepsilon^2R^6}{16}\right)\frac{\partial\psi^{(0)}}{\partial Z}\Big|_{Z_1=H_1}\\ &\quad + \frac{1}{2!}\left(\frac{\varepsilon R^4}{8} + \frac{\varepsilon^2R^6}{16}\right)^2\frac{\partial^2\psi^{(0)}}{\partial Z^2}\Big|_{Z_1=H_1} + \cdots\end{aligned}$$

Repeat this for $\psi^{(1)}(Z_1,R)$ and $\psi^{(2)}(Z_1,R)$, substitute in (4.8) and collect in ε to get

$$\psi^{(0)}(H_1,R) + \varepsilon\left[\psi^{(1)}(H_1,R) + \frac{R^4}{8}\psi_Z^{(0)}(H_1,R)\right] + \varepsilon^2\left[\psi^{(2)}(H_1,R)\right.$$
$$\left. + \frac{R^6}{16}\psi_Z^{(0)}(H_1,R) + \frac{R^4}{8}\psi_Z^{(1)}(H_1,R) + \frac{R^8}{128}\psi_{ZZ}^{(0)}(H_1,R)\right] + O(\varepsilon^3) = \frac{1}{2}R^2. \tag{4.9}$$

Similarly, the use of Taylor series expansion about the surface $Z = H_2$ gives

$$\psi^{(0)}(H_2, R) + \varepsilon \left[\psi^{(1)}(H_2, R) + \frac{R^4\kappa^3}{8}\psi_Z^{(0)}(H_2, R)\right] + \varepsilon^2 \left[\psi^{(2)}(H_2, R) + \frac{R^6\kappa^5}{16}\psi_Z^{(0)}(H_2, R) + \frac{R^4\kappa^3}{8}\psi_Z^{(1)}(H_2, R) + \frac{R^8\kappa^6}{128}\psi_{ZZ}^{(0)}(H_2, R)\right] + O(\varepsilon^3) = 0. \quad (4.10)$$

From (4.9) and (4.10), the boundary conditions for $\psi^{(i)}$ are as follows.

The boundary conditions for $\psi^{(0)}(Z, R)$ are

$$\psi^{(0)}(H_1, R) = \frac{1}{2} R^2, \quad \psi_Z^{(0)}(H_1, R) = 0 .$$

$$\psi^{(0)}(H_2, R) = 0, \quad \psi_Z^{(0)}(H_2, R) = 0 .$$

The boundary conditions for $\psi^{(1)}(Z, R)$ are

$$\begin{aligned}
\psi^{(1)}(H_1, R) &= -\frac{1}{8} R^4 \psi_Z^{(0)}(H_1, R) = 0 , \\
\psi_Z^{(1)}(H_1, R) &= -\frac{1}{8} R^4 \psi_{ZZ}^{(0)}(H_1, R) , \\
\psi^{(1)}(H_2, R) &= -\frac{1}{8} \kappa^3 R^4 \psi_Z^{(0)}(H_2, R) = 0 , \\
\psi_Z^{(1)}(H_2, R) &= -\frac{1}{8} \kappa^3 R^4 \psi_{ZZ}^{(0)}(H_2, R) .
\end{aligned}$$

The boundary conditions for $\psi^{(2)}(Z, R)$ are

$$\begin{aligned}
\psi^{(2)}(H_1, R) &= -\frac{1}{8} R^4 \psi_Z^{(1)}(H_1, R) - \frac{1}{16} R^6 \psi_Z^{(0)}(H_1, R) - \frac{1}{128} R^8 \psi_{ZZ}^{(0)}(H_1, R) \\
&= \frac{1}{128} R^8 \psi_{ZZ}^{(0)}(H_1, R) , \\
\psi_Z^{(2)}(H_1, R) &= -\frac{1}{8} R^4 \psi_{ZZ}^{(1)}(H_1, R) - \frac{1}{16} R^6 \psi_{ZZ}^{(0)}(H_1, R) - \frac{1}{128} R^8 \psi_{ZZZ}^{(0)}(H_1, R) , \\
\psi^{(2)}(H_2, R) &= \frac{1}{128} \kappa R^8 \psi_{ZZ}^{(0)}(H_2, R) , \\
\psi_Z^{(2)}(H_2, R) &= -\frac{1}{8} \kappa^3 R^4 \psi_{ZZ}^{(1)}(H_2, R) - \frac{1}{16} \kappa^5 R^6 \psi_{ZZ}^{(0)}(H_2, R) \\
&\quad -\frac{1}{128} \kappa^6 R^8 \psi_{ZZZ}^{(0)}(H_2, R) .
\end{aligned}$$

Solving (4.5) for $\psi^{(0)}$, we get

$$\psi^{(0)} = A_0 Z^3 + B_0 Z^2 + C_0 Z + D_0 ,$$

where

$$A_0 = -R^2/(H_1 - H_2)^3, \qquad B_0 = \tfrac{3}{2}R^2(H_1 + H_2)/(H_1 - H_2)^3,$$

$$C_0 = -3R^2H_1H_2/(H_1 - H_2)^3, \quad D_0 = \tfrac{1}{2}R^2H_2^2(3H_1 - H_2)/(H_1 - H_2)^3.$$

We notice that the quantity $H_1 - H_2$ occurs in each expression, and we therefore give it the name $H = H_1 - H_2$. As well as reducing the size of the solution, it has a geometrical significance, being the total distance between the two surfaces.

At the next level in the solution hierarchy, we find

$$\psi^{(1)} = -\frac{1}{10}Z^5\Upsilon A_0 - \frac{1}{6}Z^4\Upsilon B_0 + A_1Z^3 + B_1Z^2 + C_1Z + D_1,$$

where A_1, B_1, C_1,and D_1 are expressions in A_0, B_0, C_0, D_0 and Υ. At this point an important aspect of simplification, as practiced by mathematicians, becomes evident. The expression for A_1, when it is first derived, contains several terms built from the coefficients A_0 and B_0, specifically the expression is

$$A_1 = (\frac{3}{10}H_1^2 + \frac{2}{5}H_1H_2 + \frac{3}{10}H_2^2)\Upsilon A_0 + (\frac{1}{3}H_1 + \frac{1}{3}H_2)\Upsilon B_0 - \frac{1}{4}\frac{R^4(B_0 + 3A_0H_1 + \kappa^3B_0 + 3\kappa^3A_0H_2)}{H_1^2 - 2H_1H_2 + H_2^2}.$$

The second line of the above expression is simplified further by hand in the original calculation and A_1 reduces to

$$A_1 = \frac{3}{10}(H_1^2 + 4H_1H_2 + H_2^2)\Upsilon A_0 + \frac{1}{3}(H_1 + H_2)\Upsilon B_0 + \frac{3}{8}(1 - \kappa^3)R^6/H^4.$$

In some parts of this expression, the coefficients A_0 and B_0 have been simplified away, while in other parts they have been left untouched. This kind of flexibility, instinctive to a mathematician, must be allowed for in any computer algebra systems implementation. It would be difficult to anticipate this kind of simplification

automatically, so it is important to include facilities in a computer algebra systems for allowing the user to evaluate subexpressions selectively.

Examining now the structure of the solution just derived, we see that the quantities R, Z, ε and κ are the independent variables. Some systems call them atoms, but we shall call them level 0 in a hierarchy. The quantities H_1 and H_2 were defined in terms of these variables, and hence form level 1 of the hierarchy. Then H was defined at level 2, in terms of H_1 and H_2, and finally A_0, B_0, C_0,and D_0 form level 3 while A_1, B_1, C_1,and D_1 form level 4. The solutions $\psi^{(i)}$ form a parallel hierarchy which is built up with the aid of the quantities in the first hierarchy. In addition there are assorted other quantities introduced for convenience, such as the operator Υ.

4.3 Implementation

In order first to derive and then to extend the solution above, we implemented a switch-controlled evaluation process in Maple, so that the quantities defined above could be reproduced in the system. This was done by replacing expressions with procedures that would behave like the subexpressions above. For each procedure defined, a logical variable was created whose value determined whether the procedure would return simply its name, or the expression it represented. For example the quantity H_1, is represented by the procedure

```
> H1 := proc() if simplify_H1 then 1+R^2/2 else 'H1(R)' fi end:
```

We use the following skeleton for all similar procedures.

```
Subex := proc()
```

```
        if simplify_switch then
             expression_for_this_proc
        else
             'procname(args)'
        fi
    end:
```

To create a specific procedure that will hold a sub-expression in our hierarchy, we use the procedure **Make_proc**.

```
Make_proc := proc(actual_switch,actual_expression)
        actual_switch:=false:
        subs(simplify_switch=actual_switch,
        expression_for_this_proc=actual_expression,
        op(Subex)):
     end:
```

By default, we set all the simplify switches to **false**. To get the actual expression instead of the procedure name, simplify switches must be set to **true**. We now illustrate these procedures in a Maple session. The effect of the command

```
> H1 := Make_proc(simplify_H1,1+R^2/2):
```

is to create a procedure

```
> H1 := proc() if simplify_H1 then 1+R^2/2 else 'H1(R)' fi end:
```

The simplification switch for the expression H_1 is **simplify_H1**. If **simplify_H1** is set to **true**, then subsequent invocations of **H1** would produce the expression for H_1, evaluated one order in the hierarchy.

```
> simplify_H1 := true:
```

```
> H1(R);
```

$$1 + \frac{1}{2}R^2$$

If the flag is set to **false**, the invocations of **H1** just produce the unevaluated function $H_1(R)$.

```
> simplify_H1 := false:
> H1(R);
```

$$H1(R)$$

Similarly, we define H_2 and H as procedures,

```
> H2 := Make_proc(simplify_H2,kappa*R^2/2):
> H := Make_proc(simplify_H2,H1(R)-H2(R)):
```

Notice that if **simplify_H** is **true**, but **simplify_H1** and **simplify_H2** are **false**, then $H(R)$ simplifies only one level $H(R)$. To obtain a complete evaluation of $H(R)$, all lower level switches must be set to **true**.

```
> simplify_H1 := true:
> simplify_H2 := true:
> simplify_H := true:
> H(R);
```

$$1 + \frac{1}{2}R^2 - \frac{\kappa R^2}{2}$$

Solution for $\psi^{(0)}(Z, R)$. Using the Maple routine **dsolve**, we solve the partial differential equation (4.5) for $\psi^{(0)}(Z, R)$ pretending that $\psi^{(0)}$ is a function of Z only, otherwise **dsolve** will fail.

```
> de1 := diff(psi0(Z),Z$4)=0:
> dsolve(de1,psi0(Z));
```

$$\psi 0(Z) = \_C1 + \_C2Z + \_C3Z^2 + \_C4Z^3$$

```
> assign(");
```

In **dsolve**, Maple makes its own internal constants (the _C's). We want to replace Maple's internal constants by chosen names for them.

```
> psi0tmp(Z):=subs(_C4=A0(R),_C3=B0(R),_C2=C0(R),_C1=D0(R),psi0(Z)):
```

We apply the boundary conditions for $\psi^{(0)}(Z, R)$ and solve the system of equations to get the constants _C1, _C2, _C3 and _C4. The simplify switches for H_1 and H_2 are **false** (by default), so we get the expressions for these constants in terms of H1(R) and H2(R).

```
> solutionset := solve({subs(Z=H1(R), psi0(Z))=R*R/2, subs(Z=H1(R),
>                 diff(psi0(Z),Z))=0, subs(Z=H2(R),psi0(Z))=0,
>                 subs(Z=H2(R), diff(psi0(Z),Z))=0},
>                 {_C1, _C2,_C3,_C4}):
> assign(solutionset):
```

We notice that the expression $(H_1(R) - H_2(R))^3$ occurs in the denominator of these coefficients. Using $H(R) = H_1(R) - H_2(R)$, we further reduce the size of the solution.

```
> H := Make_proc(simplify_H, H1(R) - H2(R) ):
> _C4:=numer(_C4)/(simplify(subs(H1(R)=H(R)+H2(R),denom(_C4)))):
> _C3:=numer(_C3)/(simplify(subs(H1(R)=H(R)+H2(R),denom(_C3)))):
> _C2:=numer(_C2)/(simplify(subs(H1(R)=H(R)+H2(R),denom(_C2)))):
> _C1:=numer(_C1)/(simplify(subs(H1(R)=H(R)+H2(R),denom(_C1)))):
```

We create procedures for A_0, B_0, C_0 and D_0.

```
> A0:=Make_proc(simplify_A0,_C4):
> B0:=Make_proc(simplify_B0,_C3):
> C0:=Make_proc(simplify_C0,_C2):
> D0:=Make_proc(simplify_D0,_C1):
```

Simplify switches for H_1, H_2, and H are **false** (by default). In order to get the expressions for $A_0(R)$, $B_0(R)$, $C_0(R)$, and $D_0(R)$ in terms of $H_1(R)$, $H_2(R)$ and $H(R)$, simplify switches for A_0, B_0, C_0 and D_0 must be set to **true**.

```
> simplify_A0 := true:        simplify_B0 := true:
> simplify_C0 := true:        simplify_D0 := true:
> A0(R);
```

$$-\frac{R'}{H(R)^3}$$

```
> B0(R);
```

$$\frac{3}{2}\frac{R^2\,(H1(R)+H2(R))}{H(R)^3}$$

```
> C0(R);
```

$$-3\frac{R^2\,H1(R)\,H2(R)}{H(R)^3}$$

```
> D0(R);
```

$$\frac{1}{2}\frac{R^2\,H2(R)^2\,(3\,H1(R)-H2(R))}{H(R)^3}$$

Solution for $\psi^{(1)}(Z, R)$. The differential equation (4.6) for $\psi^{(1)}(Z, R)$ contains the operator Υ. The technique described above can be applied to operators as well. The procedure **Upsilon** works as an unevaluated operator when **Operate_Upsilon** is set to **false**, otherwise it operates on the argument which is a

function of R.

```
> Upsilon:=proc(Var1)
>             local term1, term2:
>             if Operate_Upsilon then
>                 term1 := diff(Var1,R$2):
>                 term2 := diff(Var1,R$1):
>                 simplify (term1 - term2/R):
>             else
>                 'procname(args)'
>             fi:
>         end:
```

Now, following the same steps as used in solving the differential equation for $\psi^{(0)}(Z, R)$, the differential equation for $\psi^{(1)}(Z, R)$ is solved subject to boundary conditions with all the simplify switches and the operating switch for Υ set to **false**.

```
> Operate_Upsilon := false:
> simplify_A0 := false:        simplify_B0 := false:
> simplify_C0 := false:        simplify_D0 := false:
```

To get the boundary conditions, we differentiate $\psi^{(0)}(Z, R)$ two times with respect to Z and then get the values at $Z = H_1(R)$ and at $Z = H_2(R)$.

```
> psi0d2 := diff(psi0(Z),Z$2):
> bc012 := subs(Z=H1(R),psi0d2):
> bc022 := subs(Z=H2(R),psi0d2):
> de2 := diff(psi1(Z),Z$4)+12*Upsilon(A0(R))*Z+4*Upsilon(B0(R))=0:
```

```
> dsolve(de2,psi1(Z));
```

$$\psi 1(Z) = -\frac{1}{6}\Upsilon(B0(R))\,Z^4 - \frac{1}{10}\Upsilon(A0(R))\,Z^5 + \_C5 + \_C6\,Z + \_C7\,Z^2 + \_C8\,Z^3$$

```
> assign(");
```

Again, we want to replace Maple's internal constants with our chosen names.

```
> psi1tmp(Z):=subs(_C8=A1(R),_C7=B1(R),_C6=C1(R),_C5=D1(R),psi1(Z)):
```

Apply the boundary conditions for $\psi^{(1)}(Z, R)$ and solve the system of equations to get the constants.

```
> solutionset1 := solve({subs(Z=H1(R),psi1(Z))=0,subs(Z=H1(R),
>          diff(psi1(Z),Z))=-(R^4/8)*bc012, subs(Z=H2(R),psi1(Z))=0,
>          subs(Z=H2(R),diff(psi1(Z),Z))= -(R^4/8)*(kappa^3)*bc022},
>          {_C5,_C6,_C7,_C8}):
> assign(solutionset1):
> collect(solutionset1,[Upsilon(A0(R)),Upsilon(B0(R))],simplify):
> _C8;
```

$$\begin{aligned}&\left(\frac{3}{10}H1(R)^2 + \frac{2}{5}H1(R)\,H2(R) + \frac{3}{10}H2(R)^2\right)\Upsilon(A0(R))\\ &\quad + \left(\frac{1}{3}H1(R) + \frac{1}{3}H2(R)\right)\Upsilon(B0(R))\\ &\quad - \frac{1}{4}\frac{R^4\,(B0(R) + 3\,A0(R)\,H1(R) + \kappa^3\,B0(R) + 3\kappa^3\,A0(R)\,H2(R))}{H1(R)^2 - 2\,H1(R)\,H2(R) + H2(R)^2}\end{aligned}$$

We notice that the constants A_0, B_0, C_0 and D_0 occur in the third operand of the constants $\_C$'s in their unevaluated form. We set the simplify switches for A_0, B_0, C_0 and D_0 to **true** only for the third operand to simplify further. After simplification, we set the switches to **false** again. This simplification demonstrates the ability of our technique to allow the user to evaluate subexpressions selectively.

```
> Coefficient1:=proc(temp)
```

```
>          local temp1,simp1,simp2,Newtemp:
>          global simplify_A0,simplify_B0,simplify_C0,simplify_D0:
>          temp1:=temp-op(3,temp):
>          simplify_A0 := true:  simplify_B0 := true:
>          simplify_C0 := true:  simplify_D0 := true:
>          op(3,temp):
>          simp1:=simplify(");
>          simplify(subs(H1(R)=H2(R)+H(R),denom(simp1))):
>          simp2:= (numer(simp1))/":
>          simplify_A0 := false:  simplify_B0 := false:
>          simplify_C0 := false:  simplify_D0 := false:
>          Newtemp:=temp1+simp2:
>     end:
```

We create procedures for A_1, B_1, C_1 and D_1.

```
> A1:=Make_proc(simplify_A1,Coefficient1(_C8)):
> B1:=Make_proc(simplify_B1,Coefficient1(_C7)):
> C1:=Make_proc(simplify_C1,Coefficient1(_C6)):
> D1:=Make_proc(simplify_D1,Coefficient1(_C5)):
```

Simplify switches for A_1, B_1, C_1 and D_1 are **false** (by default). In order to get the expression for the constants $A_1(R)$, $B_1(R)$, $C_1(R)$ and $D_1(R)$ in terms of $H_1(R)$, $H_2(R)$, $H(R)$ etc., simplify switches for these constants must be set to **true**.

```
> simplify_A1 := true:       simplify_B1 := true:
> simplify_C1 := true:       simplify_D1 := true:
```

```
> A1(R);
```

$$\left(\frac{3}{10}H1(R)^2+\frac{2}{5}H1(R)\,H2(R)+\frac{3}{10}H2(R)^2\right)\,\Upsilon(A0(R))$$
$$+\left(\frac{1}{3}H1(R)+\frac{1}{3}H2(R)\right)\,\Upsilon(B0(R))-\frac{3}{8}\frac{(-1+\kappa^3)\,R^6}{H(R)^4}$$

```
> B1(R);
```

$$\left(-\frac{1}{5}H1(R)^3-\frac{4}{5}H1(R)^2\,H2(R)-\frac{4}{5}H2(R)^2\,H1(R)-\frac{1}{5}H2(R)^3\right)\,\Upsilon(A0(R))$$
$$+\left(-\frac{1}{6}H1(R)^2-\frac{2}{3}H1(R)\,H2(R)-\frac{1}{6}H2(R)^2\right)\,\Upsilon(B0(R))$$
$$+\frac{3}{8}\frac{(2\,\kappa^3\,H1(R)-H1(R)+\kappa^3\,H2(R)-2\,H2(R))\;R^6}{H(R)^4}$$

```
> C1(R);
```

$$\frac{1}{10}\left(4\,H1(R)^2+7\,H1(R)\,H2(R)+4\,H2(R)^2\right)\;H1(R)\,H2(R)\,\Upsilon(A0(R))$$
$$+\frac{1}{3}\,(H1(R)+H2(R))\;H1(R)\,H2(R)\,\Upsilon(B0(R))$$
$$-\frac{3}{8}\frac{(\kappa^3\,H1(R)^2+2\,H1(R)\,H2(R)\,\kappa^3-2\,H1(R)\,H2(R)-H2(R)^2)\;R^6}{H(R)^4}$$

```
> D1(R);
```

$$-\frac{1}{5}\,(H1(R)+H2(R))\;H1(R)^2H2(R)^2\,\Upsilon(A0(R))-\frac{1}{6}H1(R)^2H2(R)^2\,\Upsilon(B0(R))$$
$$+\frac{3}{8}\frac{(\kappa^3\,H1(R)-H2(R))\;H1(R)\,H2(R)\,R^6}{H(R)^4}$$

We require the expressions for $\Upsilon(A_0(R))$ and $\Upsilon(B_0(R))$ which appear in $A_1(R)$, $B_1(R)$, $C_1(R)$ and $D_1(R)$. As the operator Υ differentiates with respect to R, we set the simplify switches to **true** in order to get the expressions for A_0, B_0, C_0 and D_0 in terms of R. Operating switch for Υ is also set to **true**, therefore Υ operates on the functions and differentiate partially with respect to R.

```
> Operate_Upsilon:= true:      simplify_A0 := true:
```

```
> simplify_B0 := true:          simplify_C0 := true:
> simplify_D0 := true:          simplify_H1 := true:
> simplify_H2 := true:          simplify_H := true:
>
> Upsilon(A0(R));
```

$$192\frac{R^2(-2+2\kappa+R^2-2\kappa R^2+\kappa^2 R^2)}{(-2-R^2+\kappa R^2)^5}$$

```
> Upsilon(B0(R));
```

$$-48\,R^2\,\frac{(16\kappa-8+\kappa^3 R^4+R^4-2R^2-12\kappa R^2+14\kappa^2 R^2-R^4\kappa-\kappa^2 R^4)}{(-2-R^2+\kappa R^2)^5}$$

Solution for $\psi^{(2)}(Z,R)$. The differential equation (4.7) for $\psi^{(2)}(Z,R)$ contains the operators Υ and Υ^2. We create another procedure for the operator Υ^2 as described above. The procedure `Upsilon2` works as an unevaluated operator when `Operate_Upsilon2` is set to `false`, otherwise operates twice on its argument which is a function of R.

```
> Upsilon2 := proc(Var2)
>              local term1, term2, term3:
>              if Operate_Upsilon2 then
>                 term1 := diff(Var2,R$4):
>                 term2 := diff(Var2,R$3):
>                 term3 := diff(Var2,R$2):
>                 simplify(term1- 2*term2/R +term3/R^2):
>              else
>                 'procname(args)'
```

```
>              fi:
>    end:
```

First we set all the simplify switches and operating switches for Υ and Υ^2 to **false** and get the expressions for the constants which we will get by solving (4.7) for $\psi^{(2)}(Z, R)$, in terms of $H_1(R)$, $H_2(R)$, $H(R)$, $A_0(R)$, $B_0(R)$ etc.

```
> Operate_Upsilon2:= false:   Operate_Upsilon:= false:
> simplify_A1 := false:        simplify_B1 := false:
> simplify_C1 := false:        simplify_D1 := false:
> simplify_A0 := false:        simplify_B0 := false:
> simplify_C0 := false:        simplify_D0 := false:
> simplify_H1 := false:        simplify_H2 := false:
> simplify_H  := false:
```

The differential equation for $\psi^{(2)}(Z, R)$ is given as

```
> de3 := diff(psi2(Z),Z$4) - 4*Upsilon2(A0(R))*Z^3
>      - 4*Upsilon2(B0(R))*Z^2+12*Upsilon(A1(R))*Z+4*Upsilon(B1(R))
>      + Upsilon2(A0(R))*Z^3+Upsilon2(B0(R))*Z^2
>      + Upsilon2(C0(R))*Z+ Upsilon2(D0(R)) = 0:
> dsolve(de3,psi2(Z));
```

$$\psi 2(Z) = \frac{1}{280}\Upsilon 2(A0(R))\,Z^7 + \frac{1}{120}\Upsilon 2(B0(R))\,Z^6 - \frac{1}{10}Z^5\,\Upsilon(A1(R)) - \frac{1}{120}\,Z^5\,\Upsilon 2(C0(R)) - \frac{1}{6}Z^4\,\Upsilon(B1(R)) - \frac{1}{24}\,Z^4\Upsilon 2(D0(R)) + \_C9 + \_C10\,Z + \_C11\,Z^2 + \_C12\,Z^3$$

```
> assign(''):
```

We again replace Maple's constants with the constants of our choice.

```
> psi2tmp(Z):=subs(_C12=A2(R),_C11=B2(R),_C10=C2(R)
>                                  ,_C9=D2(R),psi2(Z)):
```

Following the same steps as used above, the differential equation for $\psi^{(2)}(Z,R)$ is solved subject to boundary conditions with all the simplify switches and the operating switch for Υ and $\Upsilon 2$ set to **false**.

```
> psi1d1 := diff(psi1(Z),Z$1):    psi1d2 := diff(psi1(Z),Z$2):
> bc111 := subs(Z=H1(R),psi1d1):  bc112 := subs(Z=H1(R),psi1d2):
> bc122 := subs(Z=H2(R),psi1d2):  psi0d1 := diff(psi0(Z),Z$1):
> psi0d3 := diff(psi0(Z),Z$3):    bc011 := subs(Z=H1(R),psi0d1):
> bc013 := subs(Z=H1(R),psi0d3):  bc023 := subs(Z=H2(R),psi0d3):
```

We apply the boundary conditions for $\psi^{(2)}(Z\ R)$ and solve the system of equations to get the constants.

```
> solutionset2:=solve({subs(Z=H1(R),psi2(Z))=R^8*bc012/128,
>     subs(Z=H1(R),diff(psi2(Z),Z))=-R^4*bc112/8 - R^6*bc012/16
>     -R^8*bc013/128, subs(Z=H2(R),psi2(Z))=kappa^6*R^8*bc022/128,
>     subs(Z=H2(R),diff(psi2(Z),Z))= - kappa^3*R^4*bc122/8
>     -kappa^5*R^6*bc022/16 - kappa^6*R^8*bc023/128},
>     {_C12,_C11,_C10,_C9}):
>
> sol2:=subs([-2*H1(R)*H2(R)+H2(R)^2+H1(R)^2=H(R)^2,
>    H2(R)-H1(R)=-H(R)],collect(solutionset2,[Upsilon(A0(R)),
>    Upsilon(B1(R)),Upsilon(B0(R)),Upsilon2(A0(R)),Upsilon2(B0(R)),
>    Upsilon(A1(R)),Upsilon2(C0(R)),Upsilon2(D0(R))], simplify)):
> assign(sol2):
```

We simplify the subexpressions in the constants C_9, C_{10}, C_{11} and C_{12} using the procedure **Coefficient**.

```
> Coefficient := proc(lcl)
>    local lcl1, simp1, simp2, Newlcl:
>    global simplify_A0,simplify_B0,simplify_C0,simplify_D0:
>        lcl1:=lcl-op(11,lcl):
>        simplify_A0 := true:    simplify_B0 := true:
>        simplify_C0 := true:    simplify_D0 := true:
>        simp1:= simplify(op(11,lcl)):
>        simp2:=simplify(subs(H1(R)=H2(R)+H(R),numer(simp1)))
>                    /denom(simp1):
>        simplify_A0 := false:   simplify_B0 := false:
>        simplify_C0 := false:   simplify_D0 := false:
>        Newlcl:=lcl1+simp2:
>    end:
```

Now create procedures for $A_2(R)$, $B_2(R)$, $C_2(R)$ and $D_2(R)$.

```
> A2 := Make_proc(simplify_A2,Coefficient(_C12)):
> B2 := Make_proc(simplify_B2,Coefficient(_C11)):
> C2 := Make_proc(simplify_C2,Coefficient(_C10)):
> D2 := Make_proc(simplify_D2,Coefficient(_C9) ):
```

Simplify switches for A_2, B_2, C_2 and D_2 are **false** (by default). We set these switches to **true** to get the expressions for $A_2(R)$, $B_2(R)$, $C_2(R)$ and $D_2(R)$.

```
> simplify_A2 := true:     simplify_B2 := true:
> simplify_C2 := true:     simplify_D2 := true:
```

```
> A2(R);
```

$$\frac{1}{4}\frac{R^4\,(H1(R)^3+H2(R)^3\,\kappa^3)\,\Upsilon(A0(R))}{H(R)^2}$$
$$+\left(\frac{1}{3}H2(R)+\frac{1}{3}H1(R)\right)\,\Upsilon(B1(R))+\frac{1}{4}\frac{R^4(H2(R)^2\kappa^3+H1(R)^2)\,\Upsilon(B0(R))}{H(R)^2}$$
$$+\left(-\frac{1}{56}H2(R)^4-\frac{1}{35}H1(R)\,H2(R)^3-\frac{9}{280}H2(R)^2\,H1(R)^2\right.$$
$$\left.-\frac{1}{35}H2(R)\,H1(R)^3-\frac{1}{56}H1(R)^4\right)\,\Upsilon2(A0(R))+\left(-\frac{1}{30}H2(R)^3\right.$$
$$\left.-\frac{1}{20}H1(R)\,H2(R)^2-\frac{1}{20}H1(R)^2\,H2(R)-\frac{1}{30}H1(R)^3\right)\,\Upsilon2(B0(R))$$
$$+\left(\frac{3}{10}H2(R)^2+\frac{2}{5}H1(R)\,H2(R)+\frac{3}{10}H1(R)^2\right)\,\Upsilon(A1(R))$$
$$+\left(\frac{1}{40}H2(R)^2+\frac{1}{30}H1(R)\,H2(R)+\frac{1}{40}H1(R)^2\right)\,\Upsilon2(C0(R))$$
$$+\left(\frac{1}{12}H2(R)+\frac{1}{12}H1(R)\right)\,\Upsilon2(D0(R))-\frac{3}{4}\frac{R^6\,(\kappa^3\,H2(R)+H1(R))\,A1(R)}{H(R)^2}$$
$$-\frac{1}{4}\frac{R^4\,(\kappa^3+1)\,B1(R)}{H(R)^2}+\frac{3}{32}\frac{R^8\,(R^2-2\,\kappa^5\,H(R)+2\,H(R)+\kappa^6\,R^2)}{H(R)^5}$$

Obviously, similar expressions are obtained for B_2, C_2 and D_2. Now, we require the expressions for $\Upsilon(A_1(R))$, $\Upsilon(B_1(R))$ etc. which appear in $A_2(R)$, $B_2(R)$, $C_2(R)$ and $D_2(R)$. Again as the operator Υ differentiates with respect to R, all the simplify switches should be true.

```
> Operate_Upsilon2:= true:      Operate_Upsilon:= true:
> simplify_A0 := true:           simplify_B0 := true:
> simplify_C0 := true:           simplify_D0 := true:
> simplify_A1 := true:           simplify_B1 := true:
> simplify_C1 := true:           simplify_D1 := true:
> simplify_H1 := true:           simplify_H2 := true:
> simplify_H  := true:
>
```

```
> Upsilon(A1(R));
```

$$\frac{16}{5}R^2\left(64-3424\,\kappa\,R^2+4352\,\kappa-712\,R^2+490\,R^2\,\kappa^3+110\,R^6+8844\,\kappa^3\,R^4\right.$$
$$+220\,R^6\,\kappa-970\,R^6\,\kappa^2-13688\,\kappa^3\,R^2-1716\,R^4\,\kappa-1536\,\kappa^2\,R^4+17824\,\kappa^2\,R^2$$
$$-5696\,\kappa^2-156\,R^4-50\,R^8\,\kappa^3+35\,R^8\,\kappa^2-350\,\kappa^5\,R^6-11\,R^8\,\kappa-11\,R^8\,\kappa^5$$
$$\left.+35\,R^8\,\kappa^4+500\,\kappa^4\,R^6-5436\,\kappa^4\,R^4+R^8+\kappa^6\,R^8\right)\;/\;\left(-2-R^2+\kappa^2R\right)^7$$

Again we do not print $\Upsilon(B_1(R))$, $\Upsilon^2(A_0(R))$, $\Upsilon^2(B_0(R))$, $\Upsilon^2(C_0(R))$ and $\Upsilon^2(D_0(R))$. In this manner, we get the final solution in a form which is easier to comprehend and gives more insight to the user.

4.4 Comparison - Insight, CPU Time and Memory

A comparison of the computing resources required to compute the solution to second order in ε show that in spite of the extra overhead associated with procedure calls, both time and memory requirements were reduced by roughly 40% over the naive approach. We showed this by running the computation with all switches permanently set to true, thus producing intermediate expressions equivalent to those obtained in a standard use of Maple. Of course, the benefits multiply as the order increases. As well as the straight gain in resources, there is a gain in intelligibility of the output. This is important to a mathematician, but it is harder to measure. However, the solutions produced in this way are more compact, more readable, and offer more physical insight than the naive solutions.

4.5 Further improvements

With some more experience of using these procedures, we can improve the user interface. For the present, we note that the interface for the assignment of expression labels can be improved. The following procedure combines several steps in one. Given an expression that has been obtained during a calculation, and a subexpression that we have identified, we can replace the subexpression by its name with one call to the following procedure. In addition, we have made some other improvements. First, the use of a separate global name for each simplification switch is replaced by a global table, with one entry for each switch. Second, in anticipation of the next release of Maple, we have used the new substitution command **algsubs**. The existing **subs** command is purely syntactical, and therefore it is difficult to predict whether the command will succeed in 'seeing' a subexpression which is obvious to the user in the printout. For the purposes of demonstrating the improvements, however, we must use the existing release of Maple, and put up with the inconveniences of **subs**.

First we give the command:

```
> subsname:=proc(e:name=algebraic,f)
>          local p,n,x;  global allow_simp;
>          if nargs=3 then
>             p:= args[3];
>             if type(p,{list,set}) then p:= op(p) fi;
>          else
>             p:=NULL;
```

```
>           fi;
>           n:= lhs(e); x:= rhs(e); allow_simp(n) := false;
>           assign( n, subs( ['X'=x,'N'=n,'P'=p],
>           proc() if allow_simp(N) then X else 'N'(args) fi end)):
>           subs( x=n(p),f);
>#          algsubs( x=n(p),f);
>    end;
```

The first thing that this procedure does is create a name **e** that will contain an algebraic expression. The name **e** is equated to a procedure that does the following when called. It examines the entry **allow_simp[e]** in the global table **allow_simp**, and if the entry is **true**, the actual expression is returned. Otherwise the name **e** is returned (with arguments). Thus making the name-procedure is the first job of this procedure. Having created the name **e** and its associated procedure, this prodecure then examines expression **f** and replaces the given expression with the name it created. This procedure also sets the global table entry **allow_simp[e]** to **false**. Only the first two arguments are required. If the third argument is present, it is a list or set of arguments to the procedure we are creating. We extract the arguments from the set and put them in local variable **p**.

For example, given the expression

$$f := (1 + x^2)^2 + x + \sin(1 + x^2)$$

the effect of

```
subsname( XX= 1+x^2, f, x)
```

is to create the procedure

```
XX:= proc(x) if allow_simp(XX) then 1+x^2 else XX(x) fi end
```

to set

```
allow_simp[XX]:=false
```

and to return

```
XX(x)^2+x+ sin(XX(x))
```

Upon setting `allow_simp(XX)` to `true`, we recover the original expression. Because of the weakness of `subs`, we cannot repeat these steps with

$$g := (1 + x^2) * 2 + x + \sin(1 + x^2)$$

but this should work in the next release of Maple.

We now give a demonstration of how this new command can speed up the early stages of the computation described above. Unfortunately, we must use the current version of Maple for this demonstration, and so we have to modify our procedures a little in order to force Maple to cooperate.

```
> read'subsname.mpl':
```

First we obtain the expansions of the surfaces

```
> upper_sphere := (z-a-h)^2+r^2=a^2:
> assume(epsilon>0); assume( a>0):
> subs({z=a*epsilon*Z,h=a*epsilon,r^2=a^2*epsilon*R^2}
>                                          ,upper_sphere);
> solve(",Z):
> Z1:= simplify( "[2] );
> series( Z1,epsilon);
```

$$(1 + \frac{1}{2}R^2) + \frac{1}{8}R^4\varepsilon + \frac{1}{16}R^6\varepsilon^2 + \frac{5}{128}R^8\varepsilon^3 + \frac{7}{256}R^{10}\varepsilon^4 + O(\varepsilon^5)$$

Because Maple V Release 3 has only `subs` and not `algsubs`, we must keep the term

$1 + R^2/2$ as a group. This is done for us if we retain the series data type. If we convert to a polynomial, this does not work.

```
> Z1:=subsname( H1=1+R^2/2,simplify(''),{R});
```

$$H1(R) + \frac{1}{8}R^4\varepsilon + \frac{1}{16}R^6\varepsilon^2 + \frac{5}{128}R^8\varepsilon^3 + \frac{7}{256}R^{10}\varepsilon^4 + O(\varepsilon^5)$$

```
> lower_sphere := (z-a/kappa)^2+r^2=a^2/kappa^2:
> subs( {z=a*epsilon*Z,r^2=a^2*epsilon*R^2},lower_sphere):
> solve('',Z):
> Z2:= simplify( "[2] ):
> series( Z2,epsilon);
```

$$\frac{1}{2}R^2\kappa + \frac{1}{8}R^4\kappa^3\varepsilon + \frac{1}{16}R^6\kappa^5\varepsilon^2 + \frac{5}{128}R^8\kappa^7\varepsilon^3 + \frac{7}{256}R^{10}\kappa^9\varepsilon^4 + O(\varepsilon^5)$$

```
> Z2:=subsname( H2=kappa*R^2/2,simplify(''),{R});
```

$$Z2 := H2(R) + \frac{1}{8}R^4\kappa^3\varepsilon + \frac{1}{16}R^6\kappa^5\varepsilon^2 + \frac{5}{128}R^8\kappa^7\varepsilon^3 + \frac{7}{256}R^{10}\kappa^9\varepsilon^4 + O(\varepsilon^5)$$

In order to get Maple to solve the differential equation, we must pretend that ψ depends on Z only, otherwise Maple does not see that the partial differential equation is effectively an ordinary differential equation.

```
> eq1:= diff( pretend(Z),Z$4)=0:
> bc1:=pretend(H1(R))=R^2/2: bc2:=D(pretend)(H1(R))=0:
> bc3:=pretend(H2(R))=0: bc4:=D(pretend)(H2(R))=0:
> dsolve( {eq1,bc1,bc2,bc3,bc4},pretend(Z));
```

$$pretend(Z) = \frac{1}{2}\frac{R^2\,H2(R)^2(-3\,H1(R)+H2(R))}{\%1} + 3\,\frac{R^2\,H1(R)\,H2(R)\,Z}{\%1} - \frac{3}{2}\frac{(H2(R)+H1(R))\,R^2\,Z^2}{\%1} + \frac{R^2\,Z^3}{\%1}$$

$$\%1 := 3\,H1(R)^2\,H2(R) - H1(R)^3 - 3\,H1(R)\,H2(R)^2 + H2(R)^3$$

Maple's pretty printer notices the common denominator, but does not notice that it is a cube.

```
> subs( %1=-(H1(R)-H2(R))^3,rhs(") );
```

$$-\frac{1}{2}\frac{R^2\,H2(R)^2(-3\,H1(R)+H2(R))}{(H1(R)-H2(R))^3}-3\,\frac{R^2\,H1(R)\,H2(R)\,Z}{(H1(R)-H2(R))^3}$$
$$+\frac{3}{2}\frac{(H2(R)+H1(R))\,R^2\,Z^2}{(H1(R)-H2(R))^3}-\frac{R^2\,Z^3}{(H1(R)-H2(R))^3}$$

In addition, we as human users note a geometrical significance that H equals the total gap.

```
> tmp:=subsname(H=H1(R)-H2(R),",{R});
```

$$tmp := -\frac{1}{2}\frac{R^2\,H2(R)^2(-3\,H1(R)+H2(R))}{H(R)^3}-3\,\frac{R^2\,H1(R)\,H2(R)\,Z}{H(R)^3}$$
$$+\frac{3}{2}\frac{(H2(R)+H1(R))\,R^2\,Z^2}{H(R)^3}-\frac{R^2\,Z^3}{H(R)^3}$$

There is more structure that Maple does not see: each term is a product of a function of R and a power of Z. Again we want to capture this in order to facilitate further computation.

```
> subsname(A0=coeff(",Z,3),",{R});
```

$$-\frac{1}{2}\frac{R^2\,H2(R)^2(-3\,H1(R)+H2(R))}{H(R)^3}-3\,\frac{R^2\,H1(R)\,H2(R)\,Z}{H(R)^3}$$
$$+\frac{3}{2}\frac{(H2(R)+H1(R))\,R^2\,Z^2}{H(R)^3}-\frac{R^2\,Z^3}{H(R)^3}$$

```
> subsname(B0=coeff(",Z,2),",{R}): > subsname(C0=coeff(",Z,1),",{R}):
> subsname(D0=coeff(",Z,0),",{R});
```

$$D0(R)-3\,\frac{R^2\,H1(R)\,H2(R)\,Z}{H(R)^3}+\frac{3}{2}\frac{(H2(R)+H1(R))\,R^2\,Z^2}{H(R)^3}-\frac{R^2\,Z^3}{H(R)^3}$$

Unfortunately, Maple's **subs** command is too weak to effect the substitutions so we must wait for **algsubs**.

In view of the need to set the multiple switches to **true** or **false** during a calculation, it is useful to have a procedure to do this. We provide a procedure **switch(P,n)**, where P is a list containing the names of the functions whose entries in **allow_simp** we wish to set to **true** or **false**, depending on the value of **n** as **on/true** or **off/false** respectively.

```
> switch := proc(P:list,val:string)
>           local i:
>           global allow_simp:
>           for i in P do
>               if val = 'on' or val = 'ON' or val= 'true' then
>                    allow_simp(i) := true:
>               elif val='off' or val='OFF' or val='false' then
>                    allow_simp(i) := false:
>               fi:
>           od:
>           RETURN():
>               end:
```

For example, given the expression

```
> f:= (1+x^3)^2+sin(1+x^3)+(1+y^2)^4+cos(1+y^2)+exp(z+4)+tan(z+4):
> subsname( X = 1+x^3, f, x):
> subsname( Y = 1+y^2, '', y):
> f:=subsname( Z = z+4, '', z);
```

$$f := X(x)^2 + \sin(X(x)) + Y(y)^4 + \cos(Y(y)) + e^{Z(z)} + \tan(Z(z))$$

```
> switch([X,Y,Z],on):
```

```
> f;
```

$$(1+x^3)^2+\sin(1+x^3)+(1+y^2)^4+\cos(1+y^2)+e^{(z+4)}+\tan(z+4)$$

```
> switch([X,Y,Z],false):
> f;
```

$$X(x)^2+\sin(X(x))+Y(y)^4+\cos(Y(y))+e^{Z(z)}+\tan(Z(z))$$

```
> switch([X,Z],ON):
> f;
```

$$(1+x^3)^2+\sin(1+x^3)+Y(y)^4+\cos(Y(y))+e^{(z+4)}+\tan(z+4)$$

```
> switch([Y],true):
> switch([X,Z],OFF):
> f;
```

$$X(x)^2+\sin(X(x))+(1+y^2)^4+\cos(1+y^2)+e^{Z(z)}+\tan(Z(z))$$

Multiple substitution in a single step:

For the purpose of multiple substitution in a single step, following procedure, which is a slightly modified version of the procedure **subsname** may be used. The first argument of this procedure is a list of the substitutions in the expression **f**. Optionally, a list of the corresponding list of independent variable(s) may be given.

```
> subsnames:=proc(e:list, f)
>         local p,n,x,i,g;
>         global allow_simp;
>     g := f:
```

```
>     for i from 1 to nops(e) do
>           if nargs=3 then
>              p:= args[3];
>              if type(p,{list,set}) then
>                  p:= op(op(i,args[3])) :
>              fi;
>           else
>              p:=NULL;
>           fi;
>           n:= lhs(op(i,e)); x:= rhs(op(i,e));
>           allow_simp(n) := false;
>           assign( n,subs( ['X'=x,'N'=n,'P'=p],
>           proc() if allow_simp(N) then X else 'N'(args) fi end)):
>           g:= subs(x=n(p),g) ;
>#          g:= algsubs(x=n(p),g) ;
>     od:
> end:
```

For example, given the expression

```
> f:= (1+x^3+y^2)^2 + sin(1+x^3+y^2) + (1+u^2)^4 + cos(1+u^2)
>     + exp(z+4) + tan(z+4):
> f:=subsnames([X= 1+x^3+y^2,U= 1+u^2,Z= z+4],f,[[x,y],[u],[z]]);
```

$$f := X(x,y)^2 + \sin(X(x,y)) + U(u)^4 + \cos(U(u)) + e^{Z(z)} + \tan(Z(z))$$

```
> switch([X,U,Z],on):
```

```
> f;
```

$$(1 + x^3 + y^2)^2 + \sin(1 + x^3 + y^2) + (1 + u^2)^4 + \cos(1 + u^2) + e^{(z+4)} + \tan(z + 4)$$

```
> switch([U],false):
> f;
```

$$(1 + x^3 + y^2)^2 + \sin(1 + x^3 + y^2) + U(u)^4 + \cos(U(u)) + e^{(z+4)} + \tan(z + 4)$$

```
> switch([U],true):
> switch([X,Z],OFF):
> f;
```

$$X(x,y)^2 + \sin(X(x,y)) + (1 + u^2)^4 + \cos(1 + u^2) + e^{Z(z)} + \tan(Z(z))$$

Part II. Couples on sphere and plane using highly accurate solution of differential equation.

4.6 Introduction

It was mentioned in part I that many results for lubrication theory problems can be obtained by considering only the flow in the gap between the close surfaces. Other results require a solution for the flow outside the gap. Attempts to include the effects from outside the gap sometimes lead to unusual problems in the solution of ordinary differential equations. An example of such a problem is the calculation of the couple acting on a sphere of radius a and the couple acting on a plane wall, in the case in which the sphere moves parallel to the wall, a distance h from it. O'Neill and Stewartson (1967) studied this problem using matched inner and outer expansions, and left unsolved some questions connected with the numerical constants they obtained. The couple acting on the sphere g_s and the couple acting on the wall g_w were found to leading order to be

$$g_s = \ln(a/h) + K_1 = \ln(a/h) - 0.26227$$

and

$$g_w = \ln(a/h) + K_2 = \ln(a/h) - 0.26221.$$

The coefficient of the logarithmic terms is given solely by the gap solution, and hence can be found using the methods of part I. It is fairly easy to show that they are equal. The constants K_1 and K_2 require the determination of the flow outside the gap. These two constants are either coincidentally very close, or equal. The difficulty of the numerical problem prevented O'Neill and Stewartson (1967) from

solving it to high enough accuracy to decide one way or the other.

For fluid flow outside the narrow gap between the sphere and the plane, it is not important whether the sphere actually touches the plane or whether it is simply very close to it. For this reason, O'Neill and Stewartson (1967) approximated the flow by that for a sphere actually touching the plane. The flow field can then be expressed by an integral transform. We state without further explanation the result that the velocity field is given in cylindrical polar coordinates by $\mathcal{U}(u\cos\theta, v\sin\theta, w\cos\theta)$ and the pressure by $\mathcal{U}p\cos\theta$.

$$u = rQ + \frac{1}{2}(\psi + \chi)\,,$$

$$v = \frac{1}{2}(\chi - \psi)\,,$$

$$w = (zQ + \phi)\,,$$

$$p = \frac{2\mu Q}{a}.$$

where μ is the viscosity of the fluid, a is the radius of the sphere and ψ, ϕ, χ and Q are functions of r and z, all of the form (4.11) given below.

The coordinates (r, θ, z) are transformed to the tangent-sphere coordinates (ξ, θ, η) by

$$r = \frac{2\eta}{(\xi^2 + \eta^2)}, \qquad z = \frac{2\xi}{(\xi^2 + \eta^2)}.$$

In terms of the above coordinates the plane wall is given by $\xi = 0$, the sphere by $\xi = 1$ and the origin by $\eta = \infty$. The functions ψ, ϕ, χ and Q are then all of the form

$$f = (\xi^2 + \eta^2)^{\frac{1}{2}} \int_0^\infty \{\mathcal{A}(s)\sinh s\xi + \mathcal{B}(s)\cosh s\xi\}\, J_m(s\eta)\, ds, \qquad (4.11)$$

where $m = 0, 1, 2$. In particular

$$\phi = (\xi^2 + \eta^2)^{\frac{1}{2}} \int_0^\infty A(s) \sinh s\xi \, J_1(s\eta) \, ds.$$

The function $A(s)$ appearing in this expression is the solution of the differential equation

$$s^3 K' A'' + s A' [s^2 K'' + 3 s K' + 2 K - A [s^2 K'' + 4 s K' + 2 K] = - s^2 X'', \quad (4.12)$$

where $K = s^{-1} - \coth s$ and $X = \coth s - 1$, subject to the boundary conditions that $A(s)$ is less singular than s^{-2} at the origin and that $A(s)$ decays at infinity. The equation has a regular singular point at $s = 0$ and an irregular point at infinity.

In terms of this function, the constants K_1 and K_2 are given by

$$\begin{aligned} K_1 = \frac{4}{5} + \frac{1}{2} \int_0^\infty \Bigg\{ & \left(A + \frac{3}{5s^2}\right) [2s \operatorname{cosech}^2 s - (\coth s - 1)(1 + 2s + s^2 \operatorname{cosech}^2 s)] \\ & - \frac{3}{5} e^{-2s} \left[\frac{1}{s^3} + \frac{1}{s^2} + \frac{2}{3s} + \frac{5}{3}(2s - 1) \coth s\right] \\ & - \frac{3}{5} \left[\frac{2}{s^3} - \left(\frac{3}{s^2} + \frac{2}{s}\right)(\coth s - 1)\right] \Bigg\} \, ds, \end{aligned}$$

$$K_2 = \frac{1}{5} + \frac{1}{4} \int_0^\infty \left\{4 s A + s^2 A'' K + 2 (\coth s - 1) - 4 e^{-2s}/5s\right\} \, ds.$$

The problem is an ideal one for exploring ways in which Maple can contribute to these problems. Because of the singular points in the equation, the standard way to solve them numerically is to generate the first steps of a numerical solution using a series expansion about the origin, and about infinity. The truncation of the series and the change from one integration method to another generate some inaccuracies that might be acceptable in other contexts. However, since the question concerns the difference between the constants at the fifth and sixth significant figures, we

need a highly accurate solution. Here we show that such a solution can be obtained by Maple with relatively little effort on the part of the user, the work being done by routines that we develop in chapter 6.

The use of series expansions in the solution of differential equations has always been hampered by several difficulties. The first difficulty is the laborious nature of their derivation. This is compounded by the second ditficulty, which is the slow rate of convergence of the series. In order to obtain acceptable accuracy, many terms are needed, but the calculation of these is tedious. The final difficulty is radius of convergence. Often a singularity in the complex plane will prevent the series converging at all points of interest, although the function is well behaved on the real line.

All of these difficulties can be overcome in this case using Maple. The new routines developed in chapter 6 allow us to obtain large numbers of terms in the series with little effort and quickly. Using this fact we can re-expand the solution at different points along the axis, thus working around the convergence problem. We shall thus obtain a highly accurate solution and prove that the two constants K_1 and K_2 are indeed equal to 10 significant figures.

4.7 Solutions about $s = 0$

The differential equation (4.12) has a regular singular point at the origin and others in the complex plane at $s = i n \pi$. Unfortunately, Maple V Release 3 routine **dsolve/series** gives incorrect homogeneous solutions and no particular integral. We obtain correct homogeneous solutions and the particular integral about origin

using the routines developed in the chapter 6. We denote the homogeneous solutions about $s = 0$ as $A_h^{(0)}$ and the particular integral as $A_p^{(0)}$. Using Maple we found series correct to 75 terms.

$$A_h^{(0)} = s^{(-2+\sqrt{10})}[1 + \sum_{\substack{n=2 \\ n\,\text{even}}}^{74} a_n\, s^n],$$

where first few constants in the series are

$$\begin{aligned} a_2 &= \frac{13}{270} - \frac{2}{135}\sqrt{10}\,, \\ a_4 &= -\frac{31}{2268} + \frac{967}{226800}\sqrt{10}\,, \\ a_6 &= \frac{5}{6804} - \frac{313}{1360800}\sqrt{10}\,, \\ \ldots &\quad \ldots \quad \ldots \end{aligned}$$

The particular integral $A_p^{(0)}$, correct to 75 terms is found as

$$A_p^{(0)} = \sum_{\substack{n=-2 \\ n\,\text{even}}}^{74} b_n\, s^n,$$

where

$$\begin{aligned} b_{-2} &= -\frac{3}{5}, \qquad b_0 = -\frac{6}{25}\,, \\ b_2 &= -\frac{107}{1575}, \qquad b_4 = \frac{394}{307125}\,, \\ \ldots &\quad \ldots \quad \ldots \end{aligned}$$

We reject the other homogeneous solution because it is asymptotically $O(s^{-(2+\sqrt{10})})$, for small values of s. Hence for small s, the general solution of differential equation (4.12) is

$$A(s) = A_p^{(0)} + c\, A_h^{(0)}, \tag{4.13}$$

where the constant c has to be determined.

4.8 Series solutions about other expansion points

The solutions $A_h^{(0)}$ and $A_p^{(0)}$ converge only for $s < \pi$, a fact evident from the plots shown in Figures (4.1) and (4.2). We expand the functions $A_p^{(0)}$ and $A_h^{(0)}$ along the real axis by analytic or numerical continuation. We use the solutions $A_h^{(0)}$ and $A_p^{(0)}$ to give us boundary conditions at a new expansion point. We begin with the series solutions about $s = 5/2$, an ordinary point of the differential equation (4.12). We find the homogeneous solution $A_h^{(5/2)}$ of the homogeneous differential equation (4.12) [with X'' set to 0] subject to the boundary conditions

$$A_h^{(5/2)} = A_h^{(0)}\Big|_{s=5/2}, \qquad \frac{d}{ds} A_h^{(5/2)} = \frac{d}{ds} A_h^{(0)}\Big|_{s=5/2}.$$

Similarly, we find the particular integral $A_p^{(5/2)}$ of the nonhomogeneous differential equation (4.12) subject to the boundary conditions

$$A_p^{(5/2)} = A_p^{(0)}\Big|_{s=5/2}, \qquad \frac{d}{ds} A_p^{(5/2)} = \frac{d}{ds} A_p^{(0)}\Big|_{s=5/2}.$$

The general solution of the differential equation (4.12) about $s = 5/2$ is

$$A(s) \;=\; A_p^{(5/2)} + c\, A_h^{(5/2)}.$$

We now use these solutions to give us boundary conditions at a new expansion point $s = 4$. We find the homogeneous solution $A_h^{(4)}$ of the homogeneous differential equation subject to the boundary conditions

$$A_h^{(4)} = A_h^{(5/2)}\Big|_{s=4}, \qquad \frac{d}{ds} A_h^{(4)} = \frac{d}{ds} A_h^{(5/2)}\Big|_{s=4},$$

and the particular integral $A_p^{(4)}$ subject to the boundary conditions

$$A_p^{(4)} = A_p^{(5/2)}\Big|_{s=4}, \qquad \frac{d}{ds} A_p^{(4)} = \frac{d}{ds} A_p^{(5/2)}\Big|_{s=4}.$$

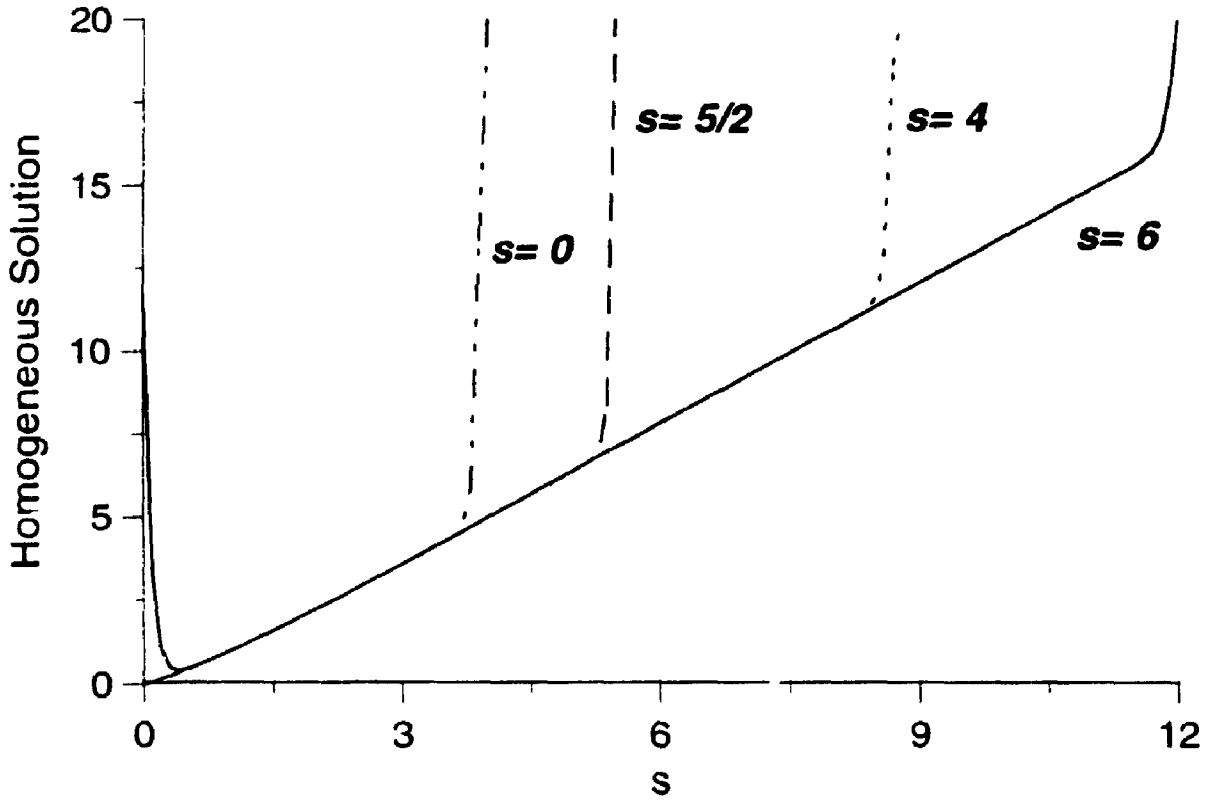

Figure 4.1: Series expansions for the homogeneous solution about various points. The expansion point is written next to each curve.

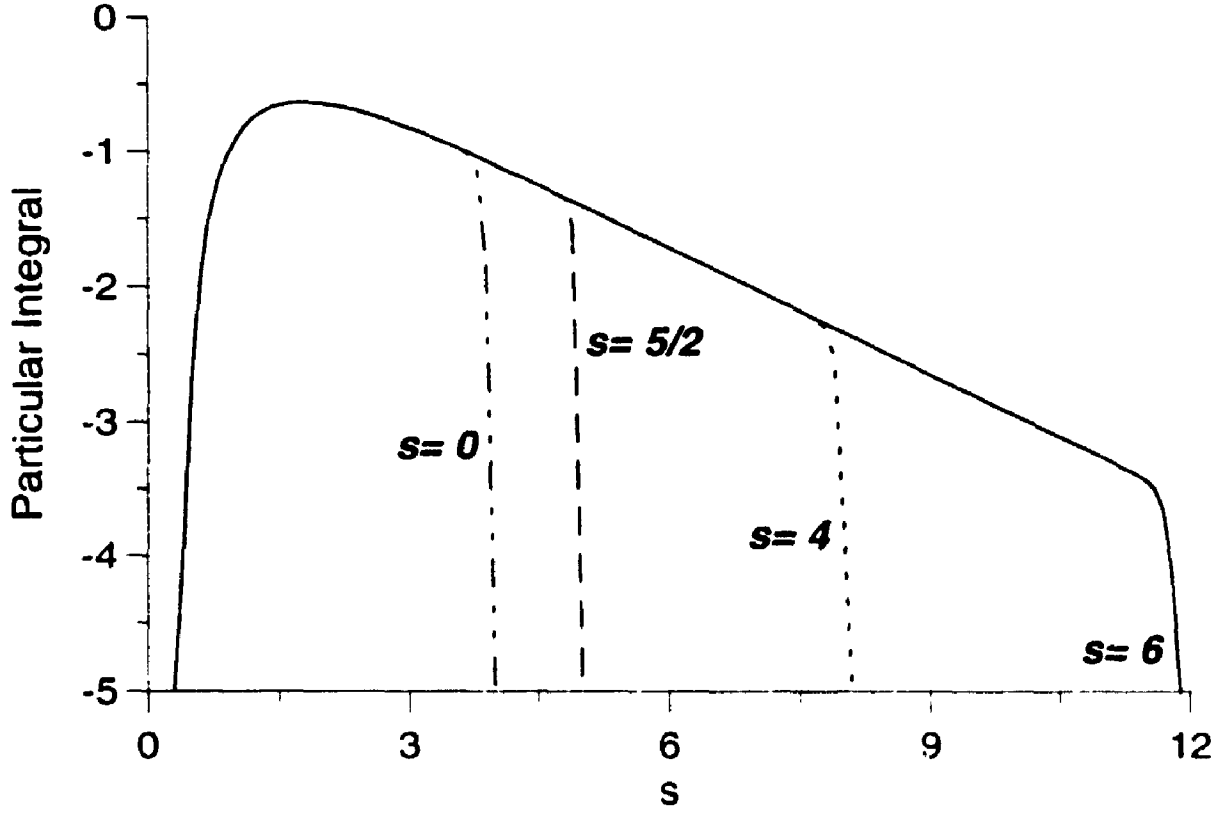

Figure 4.2: Series expansions for the particular integral about various points. The expansion point is written next to each curve.

The general solution of the differential equation (4.12) about $s = 4$ is

$$A(s) = A_p^{(4)} + c\,A_h^{(4)}.$$

Continuing in this way, we find the solutions at expansion points $s = 6$, $s = 8$ etc. To achieve high accuracy, all the solutions were found correct to 75 terms. For the series solution about an ordinary point, Maple V Release 3 routine **dsolve/series** is not efficient enough, in terms of memory requirements, to allow us to have a large number of terms in the solutions. We use the routines developed in chapter 6.

4.9 Asymptotic solution.

About $s = \infty$, we obtain an asymptotic expansion as follows.

For large s, $\coth s \sim 1$. We therefore replace $\coth s$ with 1 and find the particular integral a_p and the homogeneous solution a_h for the resulting equation to be

$$a_h = e^{-2s}, \qquad a_p = -2\,s^2\,e^{-2s}.$$

We neglect the other homogeneous solution which is $1 - 2\,s$ for large s.

For the next order approximation, $\coth s \sim 1 + 2\,e^{-2s}$. We consider the homogeneous solution a_{h2} in the form

$$a_{h2} = e^{-2s} + f(s)\,e^{-4s}.$$

Substitute a_{h2} in the differential equation (4.12), we solve the differential equation for $f(s)$ and get a_{h2} as

$$a_{h2} = e^{-2s} + 3\,s\,e^{-4s} - \frac{5}{2}\,e^{-4s} + 4\,s^2\,e^{-4s}$$

$$+6\,e^{-2s}\,\mathrm{Ei}(1,2s) + 12\,s\,\mathrm{Ei}(1,4s) - 6\,\mathrm{Ei}(1,4s).$$

Similarly, we find the particular integral a_{p2} as

$$a_{p2} \;=\; -2s^2e^{-2s} + \left(-\frac{25}{8} - \frac{25}{4}s - 10s^2 - 6s^3 - 8s^4\right)e^{-4s}.$$

Hence for large s, differential equation (4.12) has asymptotic solution

$$A(s) \;\sim\; a_{p2} + C\,a_{h2}\;, \tag{4.14}$$

where the constant C has to be determined.

4.10 Accuracy

Since the aim of this investigation is the solution of the equation (4.12) to high accuracy, we must establish the accuracy of the solutions just found. In particular, the error caused by truncating the series expansion at 75 terms must be assessed. Since we know the radius of convergence of the series about $s = 0$ is equal to π, we know that any series solution

$$\sum_{\substack{n>0\\ n\,\mathrm{even}}} a_n\,s^n$$

will have the property

$$\lim_{n\,\to\,\infty}\left|\frac{a_{2n+2}}{a_{2n}}\right| = \frac{1}{\pi^2}.$$

This is verified in the case of our solutions (See Appendix C). We can therefore estimate the error introduced by truncating the series after N terms as follows. The neglected terms will be

$$\sum_{\substack{n>N\\ n\,\mathrm{even}}} a_n\,s^n \approx \sum_{\substack{n>N\\ n\,\mathrm{even}}} a_N\,s^N\left(\frac{s}{\pi}\right)^{n-N} \;=\; a_N\,s^N\,\frac{1}{1-(s/\pi)^2}.$$

We calculate the first matching poi. . at $s = 2.5$. Hence

for $N = 50$, the error is $a_{50}\,(2.5)^{50}\,\frac{1}{1-(2.5/\pi)^2} \approx .2046055 \times 10^{-10}$, and

for $N = 74$, the error is $a_{74}\,(2.5)^{74}\,\frac{1}{1-(2.5/\pi)^2} \approx .1837556 \times 10^{-13}$.

Similar considerations apply to the series expansions about other points. Thus the series about a point $s = k$ can be written as

$$\sum_{n>0} a_n\,(s-k)^n$$

and the radius of convergence is at least k. Therefore

$$\lim_{n\,\to\,\infty}\left|\frac{a_{n+1}}{a_n}\right| \;\leq\; \frac{1}{k}.$$

From this we obtain the upper bound

$$a_{n+1} \;\leq\; a_N\left(\frac{1}{k}\right)^{n-N+1},$$

where N is the truncation order and is chosen to be large enough for the approximation to hold. Thus the error in the truncated series is

$$\begin{aligned}\sum_{n>N} a_N\left(\frac{1}{k}\right)^{n-N+1}(s-k)^{n+1} &= a_N(s-k)^N\sum_{n>N}\left(\frac{s-k}{k}\right)^{n-N+1}\\ &= a_N(s-k)^N\frac{1}{1-\frac{s-k}{k}}.\end{aligned}$$

Therefore the errors made in truncating each of the series at the point where they match the next series expansions are (given in following Table)

Expansion point	Evaluation point	Error estimate
0	5/2	$.18375568 \times 10^{-13}$
5/2	4	$.68939008 \times 10^{-15}$
4	6	$.29778618 \times 10^{-21}$
6	8	$.11522980 \times 10^{-33}$
8	10	$.22096932 \times 10^{-41}$

4.11 The Constants C and c

We find the constants C and c by matching the two solutions, for s small and for s large. The two solutions and their derivatives are equal for those values of s where these solutions are accurate. By solving the two equations, we get the constants c and C as given in Table (4.1).

N	C	c
30	3.8687549168	0.218368701355
50	3.8797383823	0.218369454539
74	3.8797393835	0.218369454587

Table 4.1: Values of constants C and c for different N.

4.12 Definite integrals K_1 and K_2

Since we have series expansions at all points, the evaluation of the integral should be equally straightforward. Unfortunately, Maple's series data structure does not accept non-rational powers, and hence this task is accomplished by replacing $(-2+\sqrt{10})$ by its rational value which is approximately 14510839/12484830. The integrals K_1 and K_2 are found as

$$K_1 = -0.2622674637,$$

$$K_2 = -0.2622674637.$$

The programs are given in Appendix B.

Chapter 5

Solution of polynomial equations in terms of independent algebraic numbers

There is no branch of mathematics, however abstract, which may not some day be applied to phenomena of the real world.

Lobachevsky

5.1 Introduction

The calculations of series solutions require the solution of polynomial equations. In particular, we need to know whether two solutions of a given polynomial differ by an integer. For simple textbook problems, this is not a difficulty, because the numerical constants are always selected so that the solutions are rational numbers. A computer-algebra system, however, must anticipate the general case in which the polynomial equation can be solved only in terms of algebraic numbers. Such solutions are represented in Maple using the `RootOf` construction, and the Maple `solve`

command will return solutions in this form. For example, given the polynomial

$$p(x) = x^{14} + 2x^{13} - 9x^{12} - 2x^{11} + 32x^{10} - 52x^9 + 221x^8 - 178x^7 + 432x^6 - 12x^5 - 223x^4 - 56x^3 + 248x^2 - 152x + 48 .$$

we can ask **solve** to find the roots. We obtain

```
> S:=[solve(p(x))];
```

$$S := [\text{RootOf}(\_Z^4 + \_Z + 1), \text{RootOf}(\_Z^4 + 8\_Z^3 + 24\_Z^2 + 40\_Z + 48), \text{RootOf}(\_Z^4 - 4\_Z^3 + 6\_Z^2 - 3\_Z + 1), 1, 1]$$

The function **RootOf** is a place holder for representing all the roots of an equation in one variable. We notice that although the equation is of degree 14, Maple has returned 5 quantities: the repeated root 1 and three **RootOf**s. Each **RootOf** stands for the four roots of the quartic polynomial that is its argument. In general, if the argument of **RootOf** is $p_1(x)$, and $p_1(x)$ is an irreducible polynomial over a field F (in our case the rationals) then $s_1 = \mathbf{RootOf}(p_1(x))$ represents an algebraic extension field K over F of degree equal to $\mathbf{degree}(p_1(x), x)$, where elements of K are represented as polynomials in s_1. The **solve** command reduces the polynomial to irreducible polynomials, and returns one **RootOf** for each polynomial.

We can evaluate a **RootOf** as a floating point number using **evalf**.

```
> evalf(op(1,S));
```

$$-.7271360845 - .4300142883I$$

We cannot control which one of the four roots is returned. We can evaluate the roots of $p(x)$ numerically. We apply **allvalues** successively to each **RootOf** and get

```
> allvalues(op(1,S));
```

$$-.7271360845 - .4300142883I, -.7271360845 + .4300142883I,$$

$$.7271360845 - .9340992895I, .7271360845 + .9340992895I$$

```
> allvalues(op(2,S));
```

$$-3.454272169 - .8600235767I, -3.454272169 + .8600285767I,$$

$$-.5457278310 - 1.868198579I, -.5457278310 + 1.868198579I$$

```
> allvalues(op(3,S));
```

$$.2728639155 - .4300142883I, .2728639155 + .4300142883I,$$

$$1.727136084 - .9340992895I, 1.727136084 + .9340992895I$$

The alert mathematician might notice that the last four values are one more than the corresponding first four. In fact, if we represent `RootOf`$(x^4 + x + 1)$ by r then the other roots are $-2 + 2r$ and $1 + r$. Clearly it is important to know this.

An inconvenience of `RootOf` in Maple V release 3 is the fact that we cannot specify which root is being referred to. Thus if we need to work individually with each root, we have no notation for this (for example we may wish to treat positive roots differently from negative ones). The research version of Maple allows `RootOf` to take a third argument which is the starting value for a search. That is to say, the internal routines that evaluate a `RootOf` numerically will start their search at the given point, and clearly if this starting point is sufficiently close to the desired root, then the user can be sure of the evaluation. Thus we can in principle distinguish the n roots of an nth degree polynomial.

Because of these observations, our aim is to take the output of a `solve` command and rework it so that the roots are separated, and any roots that differ by a rational number are noted. In particular, we wish to transfer the idea of a basis, well established in other branches of algebra, to algebraic numbers. Let a polynomial $p(x)$ have a root r that is an algebraic number, and let $p(x)$ have another root r' such that $r' = a + br$ where a and b are rational numbers. We wish to avoid expressing r' as a separate `RootOf`, or in other words, we shall say that r and r' are not linearly independent in our vector space of algebraic numbers. We wish to construct a basis for this space and express all roots in terms of it.

5.2 Theorems

Given an arbitrary polynomial, Maple can factor this to obtain a set of irreducible polynomials over the complex rationals. We therefore state the following theorems concerning sets of irreducible polynomials and the algebraic numbers they define. The proofs of these theorems are given for completeness, but they have been supplied by Drs. George Labahn, Kelly Roach and Dave Hare and do not constitute part of the original work of the thesis.

Given a polynomial $p(x)$, we assume that Maple's `factor` routine has obtained $p(x) = p_1(x)^{k_1} p_2(x)^{k_2} \ldots$, where $p_i(x)$ are irreducible polynomials over the complex rationals and k_i are integer powers.

Theorem 1. The roots of an irreducible polynomial cannot differ by a rational number, including zero.

Proof. Suppose r and $r + \rho$ are both roots of $p(x)$. Let F be the splitting field of p

over Q. Then there is an automorphism, f, of F which fixes Q and maps r to $r+\rho$ (the basic idea here is that there are automorphisms of F which fix Q and move the conjugate roots around– conjugate roots are ones which are not in the base field, Q). Since f is a field automorphism, and since f fixes Q, we must have $f(\rho) = \rho$ and $f(r) = r+\rho$ and so $f(r+\rho) = f(r) + f(\rho) = r+2\rho$. Iterating this, we see that $r + k\rho$ for all integers k must be roots of $p(x)$, which is a contradiction.

Corollary: An irreducible polynomial cannot have repeated roots, and therefore repeated roots of the original polynomial $p(x)$ can arise only because of the powers k_i.

Theorem 2. If an irreducible polynomial p_1 has a set of roots r_α and a second irreducible polynomial p_2 has a root s such that $\exists\alpha,\ s - r_\alpha = \rho$ where ρ is a rational number, then the degrees of the polynomials are equal and moreover the roots of p_2 are precisely $r_\alpha + \rho$.

Proof. Let $p_1(x)$ and $p_2(x)$ be irreducible over Q. Then the roots of $p_1(x)$ are conjugates of each other and the roots of $p_2(x)$ are conjugates of each other, but the roots of $p_1(x)$ are not conjugates of the roots of $p_2(x)$. Therefore, any automorphism of the Galois field, F, of $p(x) = p_1(x)p_2(x)$ over Q must map roots of $p_1(x)$ to roots of $p_1(x)$, roots of $p_2(x)$ to roots of $p_2(x)$ and must fix Q. So if r is a root of $p_1(x)$ and $r+\rho$ is a root of $p_2(x)$ and f is an automorphism of F which fixes Q, we must have $f(r+\rho) = f(r) + \rho$ is also a root of $p_2(x)$. Thus, since every root of $p_1(x)$ can be mapped to every other root of $p_1(x)$ by some element of the Galois group of $p(x)$. It follows that pairwise all roots of $p_1(x)$ must differ from all roots of $p_2(x)$ by the same integer, and there must be the same number of roots of each of $p_1(x)$ and $p_2(x)$.

Theorem 3. Let $p_1(x) = \sum^{N} a_n x^n$ and $p_2(x) = \sum^{N} b_n x^n$ be irreducible polynomials

that are monic. Let $\rho = (a_{N-1} - b_{N-1})/N$. If the roots of $p_1(x)$ are r_α, then the roots of $p_2(x)$ will be $r_\alpha - \rho$ iff $p_1(x + \rho) - p_2(x) = 0$.

Proof. Since $\sum r_\alpha = a_{N-1}$ the result follows.

Corollary: Two distinct irreducible polynomials cannot share a root.

Theorem 4. Given an irreducible polynomial $p_1(x)$, its roots can always be expressed in terms of a canonical polynomial $p_c(x)$ whose roots sum to zero. Specifically $p_c(x) = p_1(x - a_{N-1}/N)$.

Theorem 5. If an irreducible polynomial p_1 has a set of roots r_α and a second irreducible polynomial p_2 has a root s such that $\exists \alpha$, $s/r_\alpha = \rho$ where ρ is a rational number, then the degrees of the polynomials are equal and moreover the roots of p_2 are precisely r_α/ρ.

Theorem 6. Given two polynomials, each in canonical form (i.e. their roots sum to zero) then their roots are in the ratio ρ iff $\rho^N = a_0/b_0$ and $p_1(x) = p_2(\rho x)$.

Proof: Since the roots of $p_1(x)$ are all non-zero (since otherwise $p_1(x)$ would be reducible, having root 0), the coefficient of x^0 is the non-zero product of the roots. This coefficient is the only one that will always be non-zero. If the polynomial is of even degree, then this coefficient will not give the sign of ρ and therefore both cases must be checked.

5.3 Strategy

Based on the above, we proceed as follows. Given a polynomial $p(x)$ over the complex rationals.

1. We call **factor** to obtain $p(x) = \prod_i p_i(x)^{k_i}$.

2. We order all factors by degree and solve the linear polynomials only, to obtain the rational roots.

3. For each remaining polynomial, we use Theorem 4 to express its roots in terms of a `RootOf` based on the corresponding canonical polynomial.

4. For each pair of canonical polynomials of the same degree, we use Theorem 6 to test whether they are rational multiples of each other, and if so we express one in terms of the other.

5. For each independent `RootOf` in the basis, we ask for `allvalues` and transfer them to the `RootOf`s.

For example, we saw that

$$\begin{aligned} p(x) &= x^{14} + 2x^{13} - 9x^{12} - 2x^{11} + 32x^{10} - 52x^9 + 221x^8 - 478x^7 + 432x^6 \\ &\quad -12x^5 - 223x^4 - 56x^3 + 248x^2 - 152x + 48 \end{aligned}$$

factors into

$$p(x) = (x-1)^2(x^4 + x + 1)(x^4 + 8x^3 + 24x^2 + 40x + 48)(x^4 - 4x^3 + 6x^2 - 3x + 1)$$

The third and fourth factors are not in canonical form. Therefore we write the third factor as

$$\text{RootOf}(x^4 + 8x^3 + 24x^2 + 40x + 48) = -2 + \text{RootOf}(x^4 + 8x + 16)$$

and the fourth factor as

$$\text{RootOf}(x^4 - 4x^3 + 6x^2 - 3x + 1) = 1 + \text{RootOf}(x^4 + x + 1)$$

Since $16^{1/4} = 2$, we simplify

$$\text{RootOf}(x^4 + 8x + 16) = 2 * \text{RootOf}(x^4 + x + 1)$$

Thus the 14 roots of $p(x)$ are $1, 1, r_1, r_2, r_3, r_4, -2+2r_1, -2+2r_2, -2+2r_3, -2+2r_4, 1 + r_1, 1 + r_2, 1 + r_3, 1 + r_4$, where

$$\begin{aligned} ; &= \texttt{RootOf}(x^4 + x + 1, -.7271360845 - .4300142883I) \\ r_2 &= \texttt{RootOf}(x^4 + x + 1, -.7271360845 + .4300142883I) \\ r_3 &= \texttt{RootOf}(x^4 + x + 1, .7271360845 - .9340992895I \\ r_4 &= \texttt{RootOf}(x^4 + x + 1, .7271360845 + .9340992895I \end{aligned}$$

5.4 Implementation

We developed the procedure **Solve95** which gives a convenient solution of polynomial equations in terms of a set of algebraic extensions dependent over the rationals.

5.4.1 Solve95

Provides the solution of the polynomial using the strategy given in section (5.3)

Calling Sequence :

Solve95(S)

Parameters :

S —List of all roots, including **RootOf**'s, obtained from the routine **solve** in Maple.

Synopsis :

- The procedure finds all roots of a polynomial, reducible, irrdeucible or partially irreducible.
- It follows the strategy given in section (5.3).

Output :

Returns a convenient solution of polynomial equations in terms of a set of algebraic extensions dependent over the rationals.

Example :

$$
\begin{aligned}
P1 \;=\; & x^{22} - \frac{155}{6}x^{21} + \frac{2740}{9}x^{20} - \frac{3730193}{1728}x^{19} + \frac{211855327}{20736}x^{18} \\
& - \frac{1400911697}{41472}x^{17} + \frac{1842947915}{23328}x^{16} - \frac{10626296219}{82944}x^{15} \\
& + \frac{3571744460021}{26873856}x^{14} - \frac{5162691746987}{80621568}x^{13} - \frac{20596172148263}{967458816}x^{12} \\
& + \frac{33844025934259}{1934917632}x^{11} + \frac{158274868626865}{1934917632}x^{10} - \frac{1074629460186197}{7739670528}x^{9} \\
& + \frac{33730317592699}{859963392}x^{8} + \frac{74330144574713}{483729408}x^{7} - \frac{5748123849101041}{20639121408}x^{6} \\
& + \frac{31637395662407}{120932352}x^{5} - \frac{13338095641036993}{82556485632}x^{4} \\
& + \frac{33743127349483673}{495338913792}x^{3} - \frac{6165174337495921}{330225942528}x^{2} \\
& \frac{28238729236717}{10319560704}x - \frac{85488666082435}{990677827584}
\end{aligned}
$$

```
> S:= [solve(P1)]:

> Solve95(S);
```

COMPLETE SET OF ROOTS OF THE GIVEN POLYNOMIAL IS

$$2 + \frac{5}{4}\text{RootOf}(\_Z^4 + \_Z + 1)$$

where

$$\{\text{RootOf}(\_Z^4 + \_Z + 1, -.7271360845 + .4300142883I),$$
$$\text{RootOf}(\_Z^4 + \_Z + 1, .7271360845 + .9340992895I),$$
$$\text{RootOf}(\_Z^4 + \_Z + 1, -.7271360845 - .4300142883I),$$
$$\text{RootOf}(\_Z^4 + \_Z + 1, .7271360845 - .9340992895I)\}$$

$$-\frac{1}{2} + \frac{2}{3}\text{RootOf}(\_Z^5 + 2\_Z + 2)$$
$$\frac{2}{3} + \frac{1}{2}\text{RootOf}(\_Z^5 + \_Z^3 + 5)$$

where

$$\{\text{RootOf}(\_Z^5 + 2\_Z + 2, -.6111632249 - .9892348119I),$$
$$\text{RootOf}(\_Z^5 + 2\_Z + 2, 1.019898734 - .8770734498I),$$
$$\text{RootOf}(\_Z^5 + 2\_Z + 2, -.6111632249 + .9892348119I),$$
$$\text{RootOf}(\_Z^5 + 2\_Z + 2, 1.019898734 + .8770734498I),$$
$$\text{RootOf}(\_Z^5 + 2\_Z + 2, -.8174710190)\}$$

$$\{\text{RootOf}(\_Z^5 + \_Z^3 + 5, -.3695337942 - 1.459239192I),$$
$$\text{RootOf}(\_Z^5 + \_Z^3 + 5, .9943892280 - .8814049966I),$$
$$\text{RootOf}(\_Z^5 + \_Z^3 + 5, -.3695337942 + 1.459239192I),$$
$$\text{RootOf}(\_Z^5 + \_Z^3 + 5, .9943892280 + .8814049966I),$$
$$\text{RootOf}(\_Z^5 + \_Z^3 + 5, -1.249710867)\}$$

$$\frac{5}{2} + \text{RootOf}(\_Z^6 + \_Z + 2)$$

where

$$\{\text{RootOf}(\_Z^6 + \_Z + 2, -.9210666046 + .4532628610I),$$

$$\text{RootOf}(\_Z^6 + \_Z + 2, -.1025537771 + 1.136848701I),$$

$$\text{RootOf}(\_Z^6 + \_Z + 2, -.1025537771 - 1.136848701I),$$

$$\text{RootOf}(\_Z^6 + \_Z + 2, 1.023620382 + .6393805299I),$$

$$\text{RootOf}(\_Z^6 + \_Z + 2, 1.023620382 - .6393805299I),$$

$$\text{RootOf}(\_Z^6 + \_Z + 2, -.9210666046 - .4532628610I)\}$$

The program is given in Appendix D.

Chapter 6

Automatic Generation of Series Solutions for Ordinary Differential Equations

There was a young lady named Bright,
Whose speed was faster than light;
She set out one day
In a relative way,
And returned home the previous night.

Prof. Arthur Buller, *Punch, Dec. 19, 1923.*

6.1 Introduction

The difficulty of translating even elementary mathematical procedures into computer algebra systems is often underestimated. The series solution of ordinary differential equations is an example of this, being a topic in which existing textbooks supply only a starting point for an implementation. In particular, we note the following differences between the textbook treatment of this topic and the treatment required by a computer algebra system.

General coefficients. As we saw in our applications earlier, the coefficient functions in a user equation can be transcendental and not just simple short polynomials. Hence any study must assume that the coefficient functions will have to be expanded into series with complicated coefficients.

Solutions calculated to many terms. Since one of the attractions of computer algebra systems is their ability to work with large expressions, and at arbitrary precision, it is important that any method implemented be efficient enough to allow the calculation of a large number of terms in a series solution. Indeed, if series solutions are to be used in actual computations, as opposed to theoretical studies, large numbers of terms are indispensible.

High-order equations. It is to be expected that users will not stop with second-order equations, which form the great bulk of textbook studies. Fourth-order and sixth-order equations are quite common in research mathematics, as are large numbers of simultaneous first-order equations.

Nonhomogeneous equations. The bulk of theoretical material on the series solution of ordinary differential equations addresses only the homogeneous problem. We wish to include nonhomogeneous problems.

At least two algebra packages offer existing facilities for the series solution of ordinary differential equations: Macsyma and Maple V.

Macsyma has some capabilities for solving differential equations by series (MACSYMA reference manual, version 13, Symbolics Corp 1988). The program attempts to find the recurrence relation for the coefficients, and to solve it in closed

form. There is an option whereby an explicit series can be forced in a few cases. The package documentation explicitly rules out the class of equations containing most cases of interest to us. It states "The program does not work for expressions containing transcendentals or other functions in the coefficients of the y'', y' and y terms at this time nor does it work for nonhomogeneous equations." The current version of Macsyma (from Macsyma Inc.) has added some ability in this connection, but still cannot solve equations such as (6.3).

Maple has offered series solutions of ordinary differential equations for many years, but the package contains many unsatisfactory features which we describe in the next sections.

6.2 Errors in Maple V Release 3

The `dsolve/series` package in Maple V Release 3 attempts to obtain series solutions to an ordinary differential equation. Unfortunately, it does not always provide satisfactory solutions. We have encountered several errors in this package, which are listed below with some examples.

6.2.1 Incorrect Homogeneous Solution

In some cases, Maple V Release 3 provides incorrect homogeneous solutions to an ordinary differential equation with a regular singular point.

For example, consider the equation (Latta & Hess 1973),

$$x^2 y'' + 3x y' - (x^2 + x \coth x) y = -2x e^{-x}. \tag{6.1}$$

This equation has a regular singular point at $x = 0$, and can be solved using the method of Frobenius. The routine **dsolve/series** in Maple V Release 3 finds the series solution of (6.1) as follows

```
> de := x^2*diff(y(x),x$2)+3*x*diff(y(x),x)-(x^2+x*coth(x))*y(x)
>       = -2*x*exp(-x);
> dsolve(de,y(x),series);
```

$$y(x) = \_C1\left(\left(1+\frac{1}{9}x^2+\frac{511}{97200}x^4+O\left(x^6\right)\right)\right) + \_C2\left(\frac{\ln(x)\left(\left(-\frac{8}{3}x^2-\frac{3}{10}x^4+O\left(x^6\right)\right)\right)}{x^2}+\left(-2+\frac{7}{216}x^4+O(x^6)\right)x^{-2}\right),$$

which is wrong. The correct solutions to the homogeneous problem are $O(x^{-1+\sqrt{2}})$ and $O(x^{-1-\sqrt{2}})$ for small x.

6.2.2 Correct Homogeneous Solution, No Particular Integral

In the case of an ordinary differential equation with a regular singular point, **dsolve/series** does not provide the particular integral at all. In the following example, the differential equation has a regular singular point at the origin. Maple V Release 3 provides the correct homogeneous solutions, but no particular integral.

$$x^2y'' - xy' + (1-x)y = -2x^2e^{-x} \tag{6.2}$$

```
> de1:=x^2*diff(y(x),x$2)-x*diff(y(x),x)+(1-x)*y(x)=-2*x^2*exp(-x):
> dsolve(",y(x),series);
```

$$y(x) = \_C1x\left(1+x+\frac{1}{4}x^2+\frac{1}{36}x^3+\frac{1}{576}x^4+O(x^5)\right) + \_C2\left(x\ln(x)\left(1+x+\frac{1}{4}x^2+\frac{1}{36}x^3+\frac{1}{576}x^4+O(x^5)\right) + x\left(-2x-\frac{3}{4}x^2-\frac{11}{108}x^3-\frac{25}{3456}x^4+O(x^5)\right)\right)$$

6.2.3 Homogeneous solutions not independent

Consider the differential equation

$$x\,y'' + \cos x\,y' + y \;=\; 0 \tag{6.3}$$

The routine **dsolve/series** in Maple V Release 3 finds the series solution of (6.3) as follows

```
> de := x*diff(y(x),x$2)+cos(x)*diff(y(x),x)+y(x) = 0:
> dsolve(de,y(x),series):
> simplify(rhs(")):
```

$$\begin{aligned} y(x) \;=\; & \_C1\left(\left(1 - x + \frac{1}{4}x^2 - \frac{1}{12}x^3 + \frac{1}{48}x^4 - \frac{1}{240}x^5 + O\left(x^6\right)\right)\right) + \\ & \_C2\left(\left(1 - x + \frac{1}{4}x^2 - \frac{1}{12}x^3 + \frac{1}{48}x^4 - \frac{1}{240}x^5 + O\left(x^6\right)\right)\right) \end{aligned}$$

In the above solution, the first homogeneous solution is correct but the other is incorrect. In fact, two homogeneous solutions should be independent.

6.2.4 Incorrect homogeneous solution, no particular integral

The equation $y'''' - y = 0$ is solved correctly by Maple V Release 3, but the equation

$$y'''' - y = 1/x$$

receives an incorrect homogeneous solution, and no particular integral.

```
> diff(y(x),x$4) - y(x) = 1/x:
> dsolve(",y(x),series);
```

$$y(x) \;=\; \_C1x^3\left(1 + 1/840x^4 + O(x^6)\right) + \_C2\left(x^2\ln(x)(O(x^6))\right)$$

$$+x^2(17280 + 48x^4 + O(x^6))\Big) + \_C3\left(x\ln(x)^2(O(x^6)) + 2x\ln(x)(O(x^6))\right.$$
$$+x(-576 - 24/5x^4 + O(x^6))\Big) + \_C4\left(\ln(x)^3(O(x^6)) + 3\ln(x)^2(O(x^6))\right.$$
$$\left.+3\ln(x)(O(x^6)) + (-144 - 6x^4 + O(x^6))\right)$$

6.3 Shortcomings of Maple V Release 3

Apart from the errors described above in the existing Maple implementation, we noticed some shortcomings, which, if removed, would provide better forms of the solutions to the ordinary differential equations, or extend the scope of the package.

6.3.1 No Homogeneous Solution, No Particular Solution

If the nonhomogeneous term of a differential equation with an ordinary or regular singular point does not have a Taylor series expansion, then **dsolve/series** does not provide any solution at all. For example, the following ordinary differential equation has a regular singular point at $x = 0$,

$$x^2\,y'' + x\,y' - 4\,y \;=\; \ln x$$

```
> x^2*diff(y(x),x$2) + x*(diff(y(x),x) - 4*y(x) = ln(x);
> dsolve(",y(x),series);
```

Error, (in dsolve/series/direct) invalid arguments to coeffs.

There are other examples in which Maple V Release 3 gives a misleading error message instead of providing the correct solution. For example, for the differential equation

$$x\,y'' - \cos x\,y' + y \;=\; 0\,,$$

the routine **dsolve/series** in Maple V Release 3 gives the message as

```
> de := x*diff(y(x),x$2)-cos(x)*diff(y(x),x)+y(x) = 0:
> dsolve(de,y(x),series);
```

Error, (in dsolve/series/froben) division by zero

6.3.2 Misleading form of solutions

When the roots of the indicial equation differ by an integer, the Frobenius series solution does not always contain a logarithmic term, or equivalently the coefficient of the logarithmic term is identically zero. In any case, there is a coefficient multiplying the logarithmic term that is a constant and not a function of the independent variable. Consider the following equation

$$x^2 y'' - 4x y' + 4y = 0$$

```
> x^2*diff(y(x),x$2) -4*x*(diff(y(x),x) + 4*y(x) = 0:
> dsolve(",y(x),series);
```

$$y(x) = \_C1 x^4 \left(1 + O(x^6)\right) + \_C2(x \ln(x)(O(x^6)) + x \left(12 + O(x^6)\right)$$

In this example, the coefficient of $\ln x$ should be identically zero and, moreover, this can be easily proved. An experienced user may recognize this fact from the solution, but this form of solution is very confusing to a naive user.

6.3.3 Complex Functions instead of Real

If the indicial equation of an ordinary differential equation with a regular singular point has complex roots, then even though they are complex, they give rise to linearly independent real-valued solutions. Maple V Release 3 implementation does

not provide the real-valued solutions but prefers the complex form. For example,

```
> Order .= 4:
> (3*x^3+x^2)*diff(y(x),x$2)- x*(10*x+1)*diff(y(x),x)+(x^2+2)*y(x)=0:
> dsolve(",y(x),series);
```

$$
\begin{aligned}
y(x) \;=\; & \_C1\,x^{(1-I)}\left(1+\frac{27}{5}+\frac{19}{5}I\right)x+\left(\frac{177}{40}+\frac{769}{40}I\right)x^2 \\
& +\left(-\frac{14377}{780}+\frac{4249}{195}I\right)x^3+O(x^4)\Big) \\
& +\_C2\,x^{(1+I)}\left(1+\frac{27}{5}-\frac{19}{5}I\right)x+\left(\frac{177}{40}-\frac{769}{40}I\right)x^2 \\
& +\left(-\frac{14377}{780}-\frac{4249}{195}I\right)x^3+O(x^4)\Big).
\end{aligned}
$$

Two linearly independent real valued solutions of this differential equation are

$$
\begin{aligned}
& \_C1\,x\left[\cos(\ln(x))\left(1+\frac{27}{5}x+\frac{177}{40}x^2-\frac{14377}{780}x^3\right)\right. \\
& \qquad \left.-\sin(\ln(x))\left(-\frac{19}{5}x-\frac{769}{40}x^2-\frac{4249}{195}x^3\right)\right] \\
& +\_C2\,x\left[\cos(\ln(x))\left(-\frac{19}{5}x-\frac{769}{40}x^2-\frac{4249}{195}x^3\right)\right. \\
& \qquad \left.+\sin(\ln(x))\left(1+\frac{27}{5}x+\frac{177}{40}x^2-\frac{14377}{780}x^3\right)\right].
\end{aligned}
$$

Which form of the solution is preferred by a particular user is not predictable, but we think that mostly the purely real form will be preferred.

6.3.4 Inconsistent interface

Another shortcoming of the existing Maple V Release 3 implementation is the fact that the series solutions about an ordinary point are given in terms of unknowns $y(0)$ and $D(y)(0)$, etc., whereas the series solutions about a regular singular point are given in terms of unknowns $\_C1$, $\_C2$, etc. In the following examples, differential equation **de1** is solved about an ordinary point, $x = 0$, while **de2** is solved about

the regular singular point at origin.

```
> de1:= diff(y(x),x$2) +sin(x)*diff(y(x),x) + x*y(x) = x:
> dsolve(de1, y(x), series);
```

$$y(x) = y(0) + D(y)(0)\,x + (-1/6y(0) - 1/6D(y)(0) + 1/6)\,x^3 - 1/12D(y)(0)\,x^4 + (1/40y(0) + 1/30D(y)(0) - 1/40)\,x^5 + O(x^6)$$

In this solution, the particular integral is heavily disguised. From a pedagogical point of view, it is much more desirable to present the series in such a way that the separate homogeneous solutions and the particular integral are clearly visible to the user.

```
> de2:= x^2*diff(y(x),x$2)-2*x*diff(y(x),x)-(x^2-2)*y(x)=0:
> dsolve(de2, y(x), series);
```

$$y(x) = \_C1x^2(1 + 1/6x^2 + O(x^4)) + \_C2(x\ln(x)(O(x^4)) + x(1 + 1/2x^2 + O(x^4)))$$

6.3.5 Too much memory and time required

As is observed in previous chapters, to get the required accuracy in a solution, a differential equation has to be solved to high order. In real applications, the coefficients in the differential equations are generally complicated. In such cases, the existing package in Maple V Release 3 is not efficient.

For example, the words allocation and CPU time required to find the Frobenius series solutions for $Order = 40$ only, of the differential equation (6.1) are,

Words allocated: `status[2] := 704,383`

CPU time (sec): `status[1] := 15,516`

We shall show that the same calculations can be done using the new package using the following resources

Words allocated: `status[2] := 376,763`

CPU time (sec): `status[1] := 17`

The programs were run on a 486DX-4, 100 MHz system.

For the power series solution about an ordinary point, Maple V Release 3 is very poor in memory use. For example, the words allocation and CPU time requirement for the power series solution of (6.1) for $Order = 35$, about an ordinary point, say $5/2$, are

Words allocated: `status[2] := 3,685,725`

CPU time (sec): `status[1] := 39.94.`

The new package uses only

Words allocated: `status[2] := 212,953`

CPU time (sec): `status[1] := 1.99.`

The programs were run on a IBM RISC 6000 system. It will be shown that for more complicated equations, the computational resources in Maple V Release 3 implementation becomes worse.

6.4 Requirements of new package

In view of the above considerations, we can start with our first design decision: in which form do we return our solutions? Many simple equations, particularly those devised as problems in textbooks, can be solved by giving an explicit, and simple, recurrence relation for the coefficients in the series expansion of the solution. For

many made-up problems, these recurrence relations can even be solved in closed form and the series summed. The position taken here is that such problems are not the reason for developing this package, but rather the aim of the package is to address differential equations arising in research problems. Therefore we have decided to return only explicit series, calculated to as many terms as requested (within memory limits).

In addition to returning explicit series, we shall be designing the package to address the other issues raised above. It is assumed that the user will present an equation (or set of equations) containing coefficient functions that will need to be expanded, and that the user will ask for many terms of the series solutions. Because the second-order case is so common, it makes sense to include special code for that case whenever there is a useful gain in efficiency. Another special case that should be considered separately is that of a linear equation compared with a nonlinear one. Again the gains in efficiency, and the common occurrence of linear systems argue for special code.

A final consideration in the design of the package, one that every package must be aware of, is that returning no solution is better than returning a wrong solution. Therefore it is important to make clear, both in the theorems on which the programming is based, and in the programs themselves, exactly what class of equations can be solved by the current method.

The present program, `dsolve95`, expects as input a linear ordinary differential equation and an expansion point about which the solution is desired. A brief review of linear differential systems and the classification of the expansion point is

given in the following section.

6.5 Theoretical Background

6.5.1 Linear Differential System

The general linear nonhomogeneous ordinary differential equation is of the form

$$p_n(x)\frac{d^n}{dx^n}y(x) + p_{n-1}(x)\frac{d^{n-1}}{dx^{n-1}}y(x) + \cdots + p_0(x)\,y(x) \;=\; f(x), \qquad (6.4)$$

which may symbolically be written as

$$L(y) \;\equiv\; \left\{ p_n\,\partial^n + p_{n-1}\,\partial^{n-1} + \cdots + p_1\,\partial + p_0 \right\} y = f(x),$$

where L is a linear differential operator of order n. The coefficients p_0, p_1, p_2,$\cdots$, p_n and the nonhomogeneous term $f(x)$ are continuous, single-valued functions of x, defined on an interval $[a, b]$.

The linear differential equation (6.4) together with boundary conditions on $y(x)$ or its derivatives, forms a linear differential system. However, usually we shall have to solve (6.4) without boundary conditions, requiring the definition of a general solution. The general solution of the linear differential equation (6.4) consists of two parts:

Complementary function y_c :

Let $y_1, y_2, \ldots, y_n$ be n solutions of the homogeneous differential equation

$$L(y) \;=\; 0 \qquad (6.5)$$

associated with (6.4), such that the Wronskian W

$$W = \begin{vmatrix} y_1 & y_2 & \cdots & y_n \\ y_1' & y_2' & \cdots & y_n' \\ \vdots & \vdots & \ddots & \vdots \\ y_1^{(n-2)} & y_2^{(n-2)} & \cdots & y_n^{(n-2)} \\ y_1^{(n-1)} & y_2^{(n-1)} & \cdots & y_n^{(n-1)} \end{vmatrix} \neq 0. \tag{6.6}$$

Then the set $\{y_1\ y_2\ \cdots\ y_n\}$, called the fundamental set on the interval (a, b), is linearly independent and the linear combination

$$y_c \ = \ C_1\, y_1 + C_2\, y_2 + \cdots + C_n\, y_n$$

containing n arbitrary constants, called the complementary function, is also the solution of the homogeneous equation (6.5). The Wronskian can be calculated from the definition, or from the Abel identity, which for the system (6.4) is

$$W = W_0 \exp\left(-\int \frac{p_{n-1}}{p_n} dx\right) ,$$

where W_0 is a constant.

Particular integral y_p :

A particular integral y_p of the nonhomogeneous linear ordinary differential equation (6.4) contains no arbitrary constants. It is not, however, unique, since any complementary function can be added to it.

The general solution of (6.4) can be written as

$$y \ = \ y_c + y_p.$$

6.5.2 Classification of expansion point

From the point of view of developing the solutions of the differential equation (6.5) as infinite series, the distinction between ordinary and singular points is fundamental.

Ordinary point. The point $x = x_0$ is an ordinary point of (6.5) if **all** the functions $p_i(x)/p_n(x)$, $i = 0, 1, 2 \cdots (n-1)$, are analytic at x_0. If any one of the functions is not analytic at $x = x_0$, then the point is not an ordinary point, e.g., $x = 0$ is not an ordinary point to the differential equation

$$y'' + x^2 y' + \sqrt{x}\, y \;=\; 0,$$

because $\sqrt{x}$ is not analytic at $x = 0$.

Singular point. Let $x = x_0$ be a zero of the leading coefficient $p_n(x)$ in (6.5), then x_0 is a singular point of the differential equation if **at least one** of the functions $p_i(x)/p_n(x)$, $i = 0, 1, 2 \cdots (n-1)$ is not analytic at $x = x_0$.

6.5.3 Solution Relative to an Ordinary Point

The fundamental existence theorem (Ince 1957, §3.32) shows that if $x_0 \in [a, b]$, is an ordinary point of the differential equation (6.5), and if $p_{n-1}(x), \cdots, p_1(x)$ and $p_0(x)$ are continuous and $p_n(x)$ does not vanish at any point of that interval, then (6.5) admits a unique solution $y(x)$ which together with its first $(n-1)$ derivatives is continuous in (a, b) and satisfies the boundary conditions,

$$y(x_0) \;=\; y_0^{(0)}, \quad \partial y(x_0) = y_0^{(1)} \quad \ldots \quad \partial^{n-1} y(x_0) = y_0^{(n-1)}, \tag{6.7}$$

where $y_0^{(k)}$ are known constants. Moreover, $y(x)$ may be developed as a Taylor series about x_0, convergent in a certain interval $(x_0 - R, x_0 + R)$.

Also, if $\{y_k(x)\}$ is a set of solutions defined by the conditions

$$\partial^l y_k(x_0) = \delta_{k(l+1)}, \qquad l = 0 \ldots n-1 \qquad \text{and} \qquad k = 1 \ldots n,$$

then

$$y(x) \;=\; y_0^{(0)}\, y_1(x) + y_0^{(1)}\, y_2(x) + \cdots + y_0^{(n-1)}\, y_n(x).$$

Thus, in order to find any solution $y(x)$, it is sufficient to derive n fundamental solutions $\{y_1(x), y_2(x), \cdots, y_n(x)\}$.

6.5.4 Solution Relative to a Singular Point

The Cauchy-Euler equation,

$$x^n \frac{d^n y}{dx^n} + x^{n-1}\, a_{n-1} \frac{d^{n-1} y}{dx^{n-1}} + \cdots + x\, a_1 \frac{dy}{dx} + a_0\, y = 0,$$

where $a_0, a_1 \ldots$ are constants, has a solution of the form x^r $(x > 0)$, where r is a constant. The theory of these differential equations can be generalized to other differential equations that have special types of singularities. Here we restrict our attention to singularities of the so-called regular singular type.

Regular Singular Point. Let $x = x_0$ be a singular point of (6.5). Rewrite (6.5) in the form

$$(x - x_0)^n\, y^{(n)} + (x - x_0)^{n-1} Q_1\, y^{(n-1)} + \cdots + Q_n\, y \;=\; 0. \qquad (6.8)$$

A necessary and sufficient condition that the point $x = x_0$ is a regular singular point of (6.8) is that the functions

$$Q_i(x) = \;(x - x_0)^i\, p_{n-i}(x)/p_n(x), \qquad i = 1, 2, \ldots, n \qquad (6.9)$$

are analytic in the neighborhood of x_0.

Let $x = x_0$ be a regular singular point of (6.8), then it is possible to obtain n fundamental solutions in the neighborhood of x_0. Without loss of generality, let us suppose that $x_0 = 0$.

The method of Frobenius (described below) shows that the general form for one of the n fundamental solutions is

$$y_k(x) = \sum_j \left(\sum_{i=0}^{\infty} y_i^{(j)} x^{i+r_j} \right) (\ln x)^j \alpha_j , \qquad y_0 \neq 0 , \qquad (6.10)$$

where the quantities α_j, r_j and $y_i^{(j)}$ are to be calculated.

6.5.5 Particular Integral by Variation of Parameters

We have emphasized above that we wish to solve nonhomogeneous problems, and this requires the computation of a particular integral. One very general method for this is the Variation of parameters. Rewrite (6.4) as

$$y^{(n)} + \frac{p_{n-1}(x)}{p_n(x)} y^{(n-1)} + \cdots + \frac{p_1(x)}{p_n(x)} y' + \frac{p_0(x)}{p_n(x)} y = g(x), \qquad (6.11)$$

where

$$g(x) = \frac{f(x)}{p_n(x)} .$$

Variation of parameters gives the following formula for a particular integral y_p (Ince 1957).

Let $\{y_k(x)\}$ be the set of n independent homogeneous solutions of the differential equation (6.11) and let the Wronskian $W(y_1 \ldots y_n)$ be defined by (6.6). In addition, let the Wronskian of subsets of $n-1$ functions be defined as follows. If we omit $y_k(x)$

from the fundamental set to obtain the reduced Wronskian $W(\{y_i\}\backslash y_k)$ defined by

$$W_k = (-1)^{n-k} \begin{vmatrix} y_1 & \cdots & y_{k-1} & y_{k+1} & \cdots & y_n \\ y_1' & \cdots & y_{k-1}' & y_{k+1}' & \cdots & y_n' \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ y_1^{(n-2)} & \cdots & y_{k-1}^{(n-2)} & y_{k+1}^{(n-2)} & \cdots & y_n^{(n-2)} \\ y_1^{(n-1)} & \cdots & y_{k-1}^{(n-1)} & y_{k+1}^{(n-1)} & \cdots & y_n^{(n-1)} \end{vmatrix}, \qquad (6.12)$$

then

$$y_p(x) = \sum_{k=1}^{n} y_k \int_0^x \frac{g(t)\, W_k(t)}{W}\, dt.$$

6.6 Explicit methods for Nth order linear equations

The main issue in developing explicit solution methods for ordinary differential equations is one of efficiency. The standard algorithms that are implemented in Maple V Release 3 follow the textbook treatments, which is to lay down a procedure. Thus a textbook will 'solve' an equation by describing a procedure in the following typical terms. Assuming that the solution is of the form $\sum a_n x^n$; substitute this guess into the equation, carry out the differentiation and collect like powers of x. Equate the coefficients of like powers to obtain a set of linear equations for the unknowns a_n and solve these equations by standard methods. The advantage of implementing this procedure exactly as it is described is that the amount of mathematical work required of the programmer is small, and the Maple code is correspondingly short. The disadvantage of this approach lies in the run-time penalties incurred. Many equations can be solved on modestly endowed machines only to low order.

We shall consider nonhomogeneous linear differential equations having coefficients that are analytic within a domain $[a, b]$. We define a standard form for such

equations, and indicate how any linear differential equation can be transformed into this standard form.

6.6.1 Notation and Lemmas

Before we start, we introduce some notation, and prove some lemmas concerning it.

Definition of notation $z^{\underline{n}}$. The notation $z^{\underline{n}}$ is defined by Knuth (1992) as:

If n is a positive integer and z is a complex number, *z to the n falling* is

$$z^{\underline{n}} = z\,(z-1)(z-2)\cdots(z-n+1).$$

In other contexts this is known as the Pochhammer symbol $(z)_n$.

Lemma 1: The kth derivative of r^l is $\partial^k r^l = l^{\underline{k}} r^{l-k}$.

Lemma 2: If k is a positive integer and $l > k$, then $k^{\underline{l}} = 0$.

Lemma 3: If k is a positive integer, then $k^{\underline{k}} = k!$.

Since, in what follows, we need to multiply series together frequently, we give the following lemma.

Lemma 4: The product of two infinite Taylor series is given by

$$\sum_{n\geq 0} a_n x^n \sum_{n\geq 0} b_n x^n = \sum_{n\geq 0} \sum_{k=0}^{n} a_{n-k} b_k x^n \ .$$

Lemma 5: The product of two finite series (polynomials) is given by the following.

$$\sum_{n=N1}^{N_2} a_n x^n \sum_{m=M_1}^{M_2} b_m x^m = \sum_{p=N_1+M_1}^{N_2+M_2} \sum_{n=\max(N_1,p-M_2)}^{\min(N_2,p-M_1)} a_n b_{p-n} x^p$$

$$= \sum_{p=N_1+M_1}^{N_2+M_2} \sum_{m=\max(M_1,p-N_2)}^{\min(M_2,p-N_1)} a_{p-m} b_m x^p \ .$$

6.6.2 Standard form for equations

Let the order of the ordinary differential equation be N and let the independent variable be initially denoted by $\hat{x}$, meaning that the equation can be written in the form

$$Ly = \left[\sum_{k=0}^{N} p_k(\hat{x})\partial^k\right] y(\hat{x}) \quad = \quad F(\hat{x}) \ ,$$

where ∂ denotes differentiation. About a point $\hat{x}_0 \in [a,b]$, the coefficients p_k have Taylor series, and we write

$$p_k(\hat{x}) \quad = \quad \sum_{l\geq 0} p_l^{(k)}(\hat{x}-\hat{x}_0)^l = (\hat{x}-\hat{x}_0)^{r_k} \sum_{l\geq 0} \hat{p}_l^{(k)}(\hat{x}-\hat{x}_0)^l \ ,$$

where r_k is an integer chosen so that $\hat{p}_0^{(k)} \neq 0$. We transform this equation into a standard form by switching to $x = \hat{x} - \hat{x}_0$; we let $r = \min\limits_k r_k$, divide through by x^r and define $f(x) = F(x)/x^r$. We obtain the standard form for the class of equations we consider:

$$\sum_{k=0}^{N} x^{t_k} \sum_{l\geq 0} \hat{p}_l^{(k)} x^l \partial^k y(x) \quad = \quad f(x) \ , \tag{6.13}$$

where $\exists k, t_k = 0$.

This is the equation whose solutions we wish to express as series. In terms of this standard form, we can restate the definitions given above of ordinary and regular singular points.

Definition of Ordinary Point. Given a differential equation in the standard form (6.13), the origin is an ordinary point of the equation if $t_N = 0$.

Proof: Since $t_N = 0$, the coefficient of $\partial^N y$ is finite at the origin, and hence the functions p_i/p_N are all analytic.

Definition of regular singular point. For a differential equation in the standard

form, the origin is a regular singular point if $\forall k, \quad t_N - t_k \leq N - k$.

Proof: By the definition of Q_i in (6.9)

$$\begin{aligned} Q_k &= O(x^{N-k} x^{t_k} / x^{t_N}) \\ &= O(x^{N-k-(t_N - t_k)}). \end{aligned}$$

By the inequality, the power of x is nonnegative.

6.6.3 Homogeneous solution about an ordinary point

Theorem 1: Let the origin be an ordinary point of the equation $Ly = 0$. The solution $y(x)$ of $Ly = 0$ correct to order $O(x^{D+1})$ is given by

$$y(x) = \sum_{n=1}^{N} C_n y_n(x) = \sum_{k=0}^{D} y_k x^k, \tag{6.14}$$

where the C_n are arbitrary constants and

$$y_k = C_{k+1}/k! , \qquad 0 \leq k \leq N-1$$

$$y_k = \frac{-\sum_{i=N}^{k-1} y_i \, i^{\underline{N}} \, p_{k-i}^{(N)} - \sum_{l=0}^{N-1} \sum_{i=l}^{k-N+l} y_i \, i^{\underline{l}} \, p_{k-N+l-i}^{(l)}}{k^{\underline{N}} \, p_0^{(N)}} , \quad N \leq k \leq D \tag{6.15}$$

In addition, the series expansions of the coefficients p_k need only be continued until maximum degree $D - N$ in order to complete this calculation.

Proof: Using Lemma 1, we get $Ly = 0$ as

$$\sum_{l=0}^{N} \sum_{i \geq 0} p_i^{(l)} x^i \sum_{m \geq 0} y_m \, m^{\underline{l}} \, x^{m-l} = 0.$$

Using Lemma 4, we get

$$\sum_{l=0}^{N} \sum_{m \geq 0} \sum_{i=0}^{m} y_i \, i^{\underline{l}} \, p_{m-i}^{(l)} \, x^{m-l} = 0,$$

which is equivalent to

$$\sum_{k\geq N}\left[\sum_{l=0}^{N}\sum_{i=l}^{k-N+l} y_i\, i^{\underline{l}}\, p^{(l)}_{k-N+l-i}\right] x^{k-N} = 0.$$

Hence $\forall k \geq N$,

$$\sum_{l=0}^{N}\sum_{i=l}^{k-N+l} y_i\, i^{\underline{l}}\, p^{(l)}_{k-N+l-i} = 0.$$

Separating the Nth term from the first sum,

$$\sum_{i=N}^{k} y_i\, i^{\underline{N}}\, p^{(N)}_{k-i} + \sum_{l=0}^{N-1}\sum_{i=l}^{k-N+l} y_i\, i^{\underline{l}}\, p^{(l)}_{k-N+l-i} = 0.$$

Again break the first sum and get (since by definition $p^{(N)}_0 \neq 0$),

$$y_k\, k^{\underline{N}}\, p^{(N)}_0 + \sum_{i=N}^{k-1} y_i\, i^{\underline{N}}\, p^{(N)}_{k-i} + \sum_{l=0}^{N-1}\sum_{i=l}^{k-N+l} y_i\, i^{\underline{l}}\, p^{(l)}_{k-N+l-i} = 0,$$

which provides the required recurrence relation (6.15). In addition, we note that the greatest value of the subscript in $p^{(l)}_{k-N+l-i}$ is $D - N$.

Corollary. For $1 \leq n \leq N$,

$$y_n(x) \;=\; \frac{x^{n-1}}{(n-1)!} + \sum_{k\geq N} y_k x^k.$$

6.6.4 Particular Integral about an ordinary point

Theorem 2: Let the origin be an ordinary point of the equation $Ly = 0$. Let the function f have a series expansion of the form

$$f(x) \;=\; \sum_{l\geq 0} f_l x^l\ .$$

Then the particular integral $y_p(x)$ of $Ly = f$ correct to order $O(x^{D+1})$ is given by

$$y_p(x) \;=\; \sum_{k=N}^{D} y_k x^k,$$

where y_k are given explicitly by the following recurrence relation

$$y_k = \frac{f_{k-N} - \sum_{i=N}^{k-1} i^{\underline{N}}\, p_{k-i}^{(N)}\, y_i - \sum_{l=0}^{N-1} \sum_{i=l}^{k-N+l} i^{\underline{l}}\, y_i\, p_{k+l-N-i}^{(l)}}{k^{\underline{N}}\, p_0^{(N)}} \qquad (6.16)$$

$$N \le k \le D$$

with $y_k = 0$ for $k < N$. In addition, the series expansions of the coefficients p_k need only be continued until maximum degree $D-N$ in order to complete this calculation and the expansion of f need only be taken to degree D.

Remark: We have made the solution unique by solving the boundary value problem

$$Ly = f,\ \forall k < N,\ \partial^{k-1} y_n = 0.$$

Proof: Using the method given in the proof of Theorem 1, we obtain

$$\sum_{k \ge N} \left[\sum_{l=0}^{N} \sum_{i=l}^{k-N+l} y_i\, i^{\underline{l}}\, p_{k-N+l-i}^{(l)} \right] x^{k-N} = \sum_{k \ge N} f_{k-N}\, x^{k-N}.$$

Hence $\forall k \ge N$, we get

$$\sum_{l=0}^{N} \sum_{i=l}^{k-N+l} y_i\, i^{\underline{l}}\, p_{k-N+l-i}^{(l)} = f_{k-N},$$

which can be written as

$$y_k\, k^{\underline{N}}\, p_0^{(N)} + \sum_{i=N}^{k-1} y_i\, i^{\underline{N}}\, p_{k-i}^{(N)} + \sum_{l=0}^{N-1} \sum_{i=l}^{k-N+l} y_i\, i^{\underline{l}}\, p_{k-N+l-i}^{(l)} = f_{k-N}.$$

Hence we get the required recurrence relation (6.16).

Remark: In very simple cases, the above conventions may give solutions that the user will regard as more complicated than necessary. For example

$$y'''' - y = x$$

will be given solutions equivalent to

$$y_1 = \frac{1}{2}\cos x + \frac{1}{2}\cosh x$$

$$y_2 = \frac{1}{2}\sin x + \frac{1}{2}\sinh x$$

$$y_3 = \frac{1}{2}\cosh x - \frac{1}{2}\cos x$$

$$y_4 = \frac{1}{2}\sinh x - \frac{1}{2}\sin x$$

$$y_p = -x + \frac{1}{2}\sinh x + \frac{1}{2}\sin x$$

Examples can always be selected that make a general scheme produce a solution that is less than optimal.

Theorem 2a: Let the origin be an ordinary point of the equation $Ly = f$. Let the function f have a series expansion of the form

$$f(x) = \sum_{n \geq -\alpha} f_p x^p ,$$

where α is a positive integer such that $-\alpha < -N$. A particular integral $y_p(x)$ of $Ly = f$ correct to order $O(x^{D+1})$ is given by

$$y_p = \sum_{m=-\alpha+N}^{D} y_m^* x^m + \sum_{i=1}^{N} c_i y_i(x) \ln x ,$$

where the $y_i(x)$ are complementary functions and the coefficients y_m^* and c_i can be calculated from triangular systems.

Proof: We first show how to calculate the coefficients $-\alpha + N \leq m \leq -1$ from the coefficients f_p, where $-\alpha \leq p < -N$. Substituting in the differential equation, we obtain

$$\sum_{l=0}^{N} p^{(l)}(x)\partial^l \sum_{m \geq -\alpha+N} y_m^* x^m = \sum_{l=0}^{N} \sum_{n \geq 0} p_n^{(l)} x^n \sum_{m \geq -\alpha+N} y_m^* m^{\underline{l}} x^{m-l} .$$

Using lemma 5 and changing the definition of l to $N - l$ we get

$$\sum_{l=0}^{N} x^{l-N} \sum_{p \geq -\alpha+N} \sum_{m=-\alpha+N}^{p} y_m^* m^{\underline{N-l}} p_{p-m}^{(N-l)} x^p$$

$$= \sum_{q \geq -\alpha+N} \sum_{l=0}^{\min(q+\alpha-N,N)} \sum_{m=-\alpha+N}^{q-l} y_m^* m^{\underline{N-l}} p_{q-l-m}^{(N-l)} x^{q-N} = \sum_{p \geq -\alpha} f_p x^p.$$

Equating powers of x for powers less than $-N$ and breaking the sums into pieces as above, we obtain an equation for y_q^* for $q < -1$.

$$y_q^* q^{\underline{N}} p_0^{(N)} + \sum_{m=-\alpha+N}^{q-1} y_m^* m^{\underline{N}} p_{q-m}^{(N)} + \sum_{l=1}^{\min(q+\alpha-N,N)} \sum_{m=-\alpha+N}^{q-l} y_m^* m^{\underline{N-l}} p_{q-l-m}^{(N-l)} = f_{q-N}.$$

Having obtained these coefficients, we define a new function y_q by the equation

$$L(y_q) = \sum_{p \geq -\alpha} f_p x^p - \sum_{p=-\alpha+N}^{-1} y_p L(x^p) = \sum_{p \geq -N} \hat{f}_p x^p$$

We use this equation to calculate the coefficients c_i defined in the theorem statement. We substitute to obtain

$$\sum_{l=0}^{N} p^{(l)} \partial^l \sum_{i=1}^{N} c_i y_i \ln x = \sum_{l=0}^{N} p^{(l)} \sum_{i=1}^{N} c_i \sum_{k=0}^{l} \binom{l}{k} \partial^{l-k} y_i \partial^k \ln x$$

The case $l = k = 0$ can be removed, because this satisfies the homogeneous equation. We also interchange the order of summation.

$$\sum_{k=1}^{N} \sum_{l=k}^{N} \sum_{i=1}^{N} p^{(l)} c_i \binom{l}{k} (-1)^{k-1} (k-1)! x^{-k} \partial^{l-k} y_i$$

Since the y_i and p_i contain positive powers only, and we wish to equate negative powers of x, the process is controlled by the x^{-k} term. The only term in x^{-N} will come from the $k = N$ term, in which case $l = N$ also,

$$\sum_{i=1}^{N} p^{(N)} c_i (-1)^{N-1} (N-1)! x^{-N} y_i = f_{-N}$$

Since $y_i = x^{i-1} + O(x^N)$, only the $i = 1$ term contributes. Therefore,

$$p_0^{(N)} c_1 (-1)^{N-1} (N-1)! x^{-N} = f_{-N}$$

Proceeding thus, we equate coefficients of x^{-N+k-1} to obtain c_k. The remaining coefficients can be obtained from the previous theorem, by defining

$$y_q = \ln(x)(\sum_{i=1}^{N} c_i y_i) + y_r$$

and y_r is given by

$$L(y_r) = \sum_{l \geq -N} \hat{f}_l x^l - \sum_{i=1}^{N} c_i L(\ln(x) y_i)$$
$$= \sum_{l \geq 0} \hat{f}_l x^l$$

The problem for y_r has already been solved.

6.6.5 Homogeneous solution about a regular singular point

Theorem 1: Let the origin be a regular singular point of $Ly = 0$. Rewrite $Ly = 0$ as

$$\sum_{k=0}^{N} x^{N-k} Q_k \partial^{N-k} y(x) \;=\; 0\,,$$

where

$$Q_0(x) \;=\; 1\,,$$
$$Q_k(x) \;=\; \sum_{n \geq 0} q_n^{(k)} x^n\,,$$

then the formal solution correct to order $O(x^{D+1})$ is given by

$$y(x) \;=\; \sum_{k=0}^{D} y_k x^{k+r} \qquad \text{with} \qquad y_0 \neq 0, \tag{6.17}$$

where y_k are given explicitly by

$$y_k \;=\; -\;\frac{\sum_{l=0}^{k-1} \sum_{m=0}^{N} (r+l)^{\underline{m}} q_{k-l}^{(N-m)} y_l}{F(r+k)}\,, \qquad k \geq 1 \tag{6.18}$$

and r is the solution of the equation

$$\sum_{m=0}^{N} r^{\underline{m}}\, q_0^{(N-m)}, \tag{6.19}$$

which is called indicial equation.

Proof: mth term in $Ly = 0$ is

$$x^m\, Q_{N-m}(x)\, y^{(m)}(x),$$

where $y^{(m)}(x)$ is mth derivative of $y(x)$.

Let

$$Q_{N-m}(x) \;=\; \sum_{n\geq 0} q_n^{(N-m)}\, x^n,$$

and

$$y(x) \;=\; \sum_{l\geq 0} y_l\, x^{l+r}.$$

Using notation given in section(6.6.1) and Lemma 1, $Ly = 0$ gives

$$\sum_{m=0}^{N} x^m \sum_{n\geq 0} q_n^{(N-m)}\, x^n \sum_{l\geq 0} (l+r)^{\underline{m}}\, y_l\, x^{l+r-m} \;=\; 0.$$

Using Lemma 4, we get

$$\sum_{m=0}^{N} \sum_{k\geq 0} \sum_{l=0}^{k} (l+r)^{\underline{m}}\, q_{k-l}^{(N-m)}\, y_l\, x^k \;=\; 0.$$

Hence $\forall \quad k \geq 0$,

$$\sum_{l=0}^{k} \sum_{m=0}^{N} (l+r)^{\underline{m}}\, q_{k-l}^{(N-m)}\, y_l \;=\; 0.$$

For $k = 0$, we get

$$F(r) = \sum_{m=0}^{N} r^{\underline{m}}\, q_0^{(N-m)}\, y_0 \;=\; 0, \tag{6.20}$$

since it is assumed that $y_0 \neq 0$,

$$\sum_{m=0}^{N} r^{\underline{m}} q_0^{(N-m)} = 0,$$

which is the indicial equation. For $k \geq 1$, we get

$$y_k \sum_{m=0}^{N} (k+r)^{\underline{m}} q_0^{(N-m)} + \sum_{l=0}^{k-1} \sum_{m=0}^{N} (l+r)^{\underline{m}} q_{k-l}^{(N-m)} y_l = 0.$$

From (6.20)

$$\sum_{m=0}^{N} (r+k)^{\underline{m}} q_0^{(N-m)} = F(r+k),$$

which gives

$$y_k F(r+k) + \sum_{l=0}^{k-1} \sum_{m=0}^{N} (l+r)^{\underline{m}} q_{k-l}^{(N-m)} y_l = 0.$$

Hence we get the required recurrence relation (6.18).

Solutions corresponding to the N roots of the indicial equation

(Ince 1957; Della Dorra & Tournier 1981).

If N roots of the indicial equation are distinct and no two of them differ by an integer then for each root r, y_k in (6.18) can be obtained and hence N distinct solutions corresponding to N distinct roots of the indicial equation form a fundamental set. Otherwise, the roots may be arranged as follows.

$$\begin{array}{cccc} r_0 & r_1 & \cdots & r_{\alpha-1} \\ r_\alpha & r_{\alpha+1} & \cdots & r_{\beta-1} \\ \cdots & \cdots & \cdots & \cdots \end{array}$$

For the first row we have,

$$\mathrm{Re}(r_0) \geq \mathrm{Re}(r_1) \geq \cdots \geq \mathrm{Re}(r_{\alpha-1})$$

$$r_j - r_i \quad \text{is an integer for} \quad j \leq i \quad \text{and} \quad 0 \leq i, \quad j \leq \alpha - 1,$$

and likewise the other rows. In each row, the multiple roots can be distinguished from the others, e.g., for the first row,

$$\begin{aligned} r_0 &= r_1 = \cdots = r_{i-1} && \text{roots of multiplicity } i \\ r_i &= r_{i+1} = \cdots = r_{j-1} && \text{roots of multiplicity } j-i \\ \cdots & \qquad \cdots \\ r_k &= r_{k+1} = \cdots = r_{\alpha-1} && \text{roots of multiplicity } \alpha-k, \end{aligned}$$

and similar for the other rows. Consider the solutions corresponding to first row of the above array. The other rows can be treated similarly. The solution associated with the root with largest real part can be obtained from (6.18). But for rest of the roots, this method fails because of the following reasons:

- For the equal roots, the expressions for the solutions are same.
- For the roots with integer difference, the denominator in (6.18) will be zero after certain rank, e.g., if two roots of the indicial equation $F(r) = 0$ are r_n and $r_n + 3$, then for $k = 3$, the denominator in (6.18) will be $F(r_n + 3)$, which is zero because $r_n + 3$ is a root of the indicial equation.

In order to avoid this, y_0 in (6.17) is replaced by the product

$$y_0 \prod_{i=1}^{w} F(r+i)\,, \qquad\qquad w = r_0 - r_{\alpha-1},$$

then

$$y(x) = x^r \left[y_0 \prod_{i=1}^{w} F(r+i) + \sum_{k\geq 1} y_k\, x^k \right],$$

and since $F(r) = 0$, $Ly = 0$ can be written as

$$Ly = y_0 \prod_{i=0}^{w} F(r+i)\, x^r.$$

The operator L and $\partial^s/\partial r^s$ are permutable, hence

$$L(\frac{\partial^s}{\partial r^s} y(x)) = \frac{\partial^s}{\partial r^s}(y_0 \prod_{i=0}^{w} F(r+i)\, x^r).$$

If we denote the product $y_0\, F(r)\, F(r+1) \cdots F(r+w)\, x^r$ as $f(r)$, then

$$\frac{\partial^s}{\partial r^s}(y_0\, f(r)\, x^r) \;=\; y_0\,[f^{(s)}(r)x^r + r \ln x f^{(s-1)}(r)x^r + \cdots + r^s x^r f(r)].$$

Hence for the first subset

$$r_0 = r_1 = \cdots = r_{i-1},$$

where r_0 is a root of order i,

$$\frac{\partial^s}{\partial r^s}(f(r)\, x^r)\Big|_{r=r_0} \;=\; 0$$

implies that

$$L(\frac{\partial^s}{\partial r^s}\, x^r \sum_{i\geq 0} y_i\, x^i\Big|_{r=r_0}) \;=\; 0 \qquad \text{for } 0 \leq s \leq i-1.$$

This shows that the formal series:

$$\frac{\partial^s}{\partial r^s}\, x^r \sum_{i\geq 0} y_i\, x^i\Big|_{r=r_0} \qquad \text{for } 0 \leq s \leq i-1$$

are formal solutions of $Ly = 0$.

Remark: In some cases, the form of the solution obtained from this method may differ from that obtained from the direct substitution (described in section [6.7] for 2nd order differential equations). Both forms of the solution are correct.

Let $y_1(x)$ and $y_2(x)$ be two linearly independent solutions obtained by using this method and let $y_1^*(x)$ and $y_2^*(x)$ be two linearly independent solutions obtained by the other method (section [6.7]). Then we can write

$$y_2(x) = K_1\, y_2^*(x) + K_2\, y_1(x),$$

where K_1 and K_2 are constants. For example, consider the differential equation with $x = 0$ as a regular singular point

$$x\, y'' + (x-1)\, y' - 2y = 0.$$

The solution obtained by this method is

$$\begin{aligned} y_1(x) &= x^2 \\ y_2(x) &= (2\ln(x)x^2 - 2 + 4x - 3x^2 - \frac{2}{3}x^3 + \frac{1}{12}x^4 - \frac{1}{90}x^5 + \cdots). \end{aligned}$$

The solution obtained by the method in section (6.7) is

$$\begin{aligned} y_1^*(x) &= x^2 \\ y_2^*(x) &= (-\ln(x)x^2 + 1 - 2x + \frac{1}{3}x^3 - \frac{1}{24}x^4 + \frac{1}{180}x^5 - \cdots). \end{aligned}$$

It can easily be verified that

$$y_2(x) = -2\, y_2^*(x) - 3\, y_1(x)\,.$$

6.6.6 Particular integral about a regular singular point

Theorem 1: Let the origin be a regular singular point of $Ly = 0$. Rewrite $Ly = F(x)$ as

$$\sum_{k=0}^{N} x^{N-k} Q_k\, \partial^{N-k} y(x) = f(x),$$

where

$$\begin{aligned} Q_0(x) &= 1\,, \\ Q_k(x) &= x^k \frac{p_{N-k}(x)}{p_N(x)}\,, \qquad 1 \le k \le N\,, \\ f(x) &= \frac{F(x)}{p_N(x)}\, x^N\,. \end{aligned}$$

Since the origin is a regular singular point of $Ly = 0$, $Q_k(x)$, $0 \le k \le N$ are analytic functions at origin. Let

$$Q_k(x) = \sum_{j=0}^{\gamma(k)} q_j^{(k)} x^{j+t_k}$$

where t_k is the lowest degree of Q_k, $1 \le k \le N$.

Let the function $f(x)$ have a series expansion of the form

$$f(x) = \sum_{i=a}^{\beta} f_i x^i.$$

Then the particular integral y_I correct to order $O(x^{D+1})$ given by

$$y_I = \sum_{i=\alpha}^{D} y_i\, x^i$$

is unique if

$$\alpha = a - \min_k \{t_k\}\,, \qquad \beta = D + \min_k \{t_k\}\,.$$

The coefficient Q_k need only be continued until maximum degree

$$\gamma(k) = D - a + 2\left[\min_k \{t_k\}\right].$$

Proof: Using Lemma 1, $Ly = f$ can be written as

$$\sum_{m=0}^{N} \sum_{k=0}^{\gamma(k)} \sum_{i=\alpha}^{D} i^{\underline{m}}\, q_k^{(N-m)}\, y_i\, x^{i+k+t_k} = \sum_{i=a}^{\beta} f_i\, x^i. \tag{6.21}$$

To compare the lowest power of x in both sides, we must have

$$\alpha = a - \min_k \{t_k\}.$$

y_I contains $D - \alpha$ coefficients. To find these coefficients uniquely, we require $D - \alpha$ equations to be solved. Thus we require

$$\begin{aligned} \beta &= D - \alpha + a \\ &= D + \min_k \{t_k\} \end{aligned}$$

For $1 \leq m \leq N$, the maximum degree of left hand side in (6.21) must be β. Hence the series expansion of the coefficients Q_k need only be continued until maximum degree

$$\gamma(k) = D - 2\alpha + a = D - a + 2\left[\min_k \{t_k\}\right].$$

This theorem is not applicable for the Nth order differential equation with equal roots or roots with integer difference.

Theorem 2: Let x_0 be a regular singular or ordinary point of $Ly = 0$. If the function f does not have Taylor series expansion about x_0, we can obtain a formal expression for a particular integral y_I for the nonhomogeneous equation $Ly = f$ by variation of parameters.

6.7 Explicit methods for second order equations

We now reconsider the results of the previous section in order to modify them so as to take advantage of the special properties of second order nonhomogeneous linear differential equations. The case of ordinary points requires only the substitution $N = 2$ in the above formulae, so we proceed to consider a regular singular point.

Theorem 1: Let the origin be a regular singular point of the second order homogeneous linear differential equation $Ly = 0$. Rewrite $Ly = 0$ as

$$\sum_{k=0}^{2} x^{2-k} Q_k \partial^{2-k} y(x) \;=\; 0\,, \tag{6.22}$$

where

$$\begin{aligned} Q_0(x) &= 1\,, \\ Q_k(x) &= x^k \frac{p_{2-k}(x)}{p_2(x)}, \qquad k = 1, 2\,. \end{aligned}$$

Since the origin is a regular singular point, $Q_1(x)$ and $Q_2(x)$ are analytic at origin. Let

$$Q_k(x) \;=\; \sum_{i \geq 0} q_i^{(k)} x^i$$

and let $y_1(x)$ and $y_2(x)$ be two fundamental Frobenius series solutions related to the origin. Then the first solution $y_1(x)$ corresponding to the larger root r_1 of the indicial equation,

$$F(r) \;=\; r(r-1) + q_0^{(1)} r + q_0^{(2)} = 0$$

correct to $O(x^{D+1})$ is given by

$$y_1(x) \;=\; \sum_{n=0}^{D} y_n^{(1)} x^{n+r_1}, \qquad\qquad y_0^{(1)} \neq 0$$

where

$$y_n^{(1)} \;=\; \frac{-\sum_{j=0}^{n-1} \left[(r_1 + j) q_{n-j}^{(1)} + q_{n-j}^{(2)} \right] y_j^{(1)}}{F(r_1 + n)}\,, \qquad n \geq 1\,. \tag{6.23}$$

The second linearly independent Frobenius series solution $y_2(x)$ corresponding to the second root r_2 of the indicial equation can be found as follows.

$r_1 - r_2 \neq 0$ and not an integer

The second solution $y_2(x)$ correct to $O(x^{D+1})$ is then given by

$$y_2(x) = \sum_{n=0}^{D} y_n^{(2)} x^{n+r_2}, \qquad y_0^{(2)} \neq 0 ,$$

where

$$y_n^{(2)} = \frac{-\sum_{j=0}^{n-1} \left[(r_2 + j) q_{n-j}^{(1)} + q_{n-j}^{(2)}\right] y_j^{(2)}}{F(r_2 + n)} , \qquad n \geq 1 . \tag{6.24}$$

$r_1 - r_2 = 0$

The second linearly independent solution $y_2(x)$ correct to $O(x^{D+1})$ is given by

$$y_2(x) = y_1(x) \ln x + \sum_{n=1}^{D} y_n^* x^{n+r_1} ,$$

where

$$y_1^* = \frac{-2 y_1^{(1)} - q_1^{(1)} y_0^{(1)}}{F(r_1 + 1)} ,$$

$$y_n^* = \frac{-\left[2n y_n^{(1)} + \sum_{j=1}^{n} q_j^{(1)} y_{n-j}^{(1)} + \sum_{j=1}^{n-1} \left\{(r_1 + j) q_{n-j}^{(1)} + q_{n-j}^{(2)}\right\} y_j^*\right]}{F(r_1 + n)} ,$$

$$n \geq 2 .$$

$r_1 - r_2 =$ positive integer N

The second Frobenius series solution $y_2(x)$ correct to $O(x^{D+1})$ is given by

$$y_2(x) = K y_1(x) \ln x + \sum_{n=0}^{D} y_n^* x^{n+r_2} ,$$

where

$$y_n^* = \frac{-\sum_{j=0}^{n-1} \left[(r_2 + j) q_{n-j}^{(1)} + q_{n-j}^{(2)}\right] y_j^*}{F(r_2 + n)} , \qquad 1 \leq n \leq N - 1$$

$$K = \frac{-\sum_{j=0}^{N-1} \left[(r_2 + j) q_{N-j}^{(1)} + q_{N-j}^{(2)}\right] y_j^*}{N} ,$$

$$y_n^* = \frac{-K\left[(2n-N)y_{n-N}^{(1)} + \sum_{j=1}^{n-N} q_j^{(1)} y_{n-N-j}^{(1)}\right] - \sum_{j=0}^{n-1}\left[(r_2+j)q_{n-j}^{(1)} + q_{n-j}^{(2)}\right] y_j^*}{F(r_2+n)},$$

$$n \geq N+1 .$$

Remark: The coefficient of y_N^* is $F(r_1)$, which is zero.

Implementation: Always compute series at least until K is determined even if user misguidedly asks for lower accuracy.

Proof:

$\underline{r_1 - r_2 \neq 0 \text{ and not an integer}}$

Substituting $y_1(x)$ and $y_2(x)$ in the differential equation and comparing the coefficients of like powers of x, we get (6.23) and (6.24).

$\underline{r_1 - r_2 = 0}$

The indicial equation

$$F(r) = r^2 + r(q_0^{(1)} - 1) + q_0^{(2)} = 0$$

has repeated roots if

$$(q_0^{(1)} - 1)^2 - 4q_0^{(2)} = 0.$$

Hence, we get

$$r_1 = r_2 = \frac{1 - q_0^{(1)}}{2} . \tag{6.25}$$

The first solution, corresponding to r_1, is given by (6.23). The second solution is of the form

$$y_2(x) = y_1(x) \ln x + Y_2(x), \tag{6.26}$$

where $Y_2(x)$ correct to $O(x^{D+1})$ is given by

$$Y_2(x) = \sum_{n=1}^{D} y_n^* x^{n+r_1} .$$

To find y_n^* explicitly, we substitute (6.26) in (6.22) and get

$$x^2 Y_2''(x) + xQ_1(x)Y_2'(x) + Q_2(x)Y_2(x) = -2xy_1'(x) - (Q_1(x) - 1)\, y_1(x).$$

Using the series expansion for $Q_1(x)$, $Q_2(x)$ and (6.25), the coefficient of x^r in right hand side becomes zero and we get

$$F(r_1+1)\, y_1^* x^{r_1+1} + \sum_{n\geq 2} \left[F(r_1+n)\, y_n^* + \sum_{j=1}^{n-1} \left\{ (r_1+j)\, q_{n-j}^{(1)} + q_{n-j}^{(2)} \right\} y_j^* \right] x^{r_1+n}$$
$$= \left[-2\, y_1^{(1)} - q_1^{(1)}\, y_0^{(1)} \right] x^{r1+1} - \sum_{n\geq 2} \left[2n\, y_n^{(1)} + \sum_{j=1}^{n} q_j^{(1)}\, y_{n-j}^{(1)} \right] x^{r_1+n}.$$

Comparing the like powers of x, we get the required recurrence relation to find y_n^* explicitly.

<u>$r_1 - r_2$ = positive integer N</u>

Let the roots r_1 and r_2 of the indicial equation be such that $r_1 > r_2$ and $r_1 - r_2$ is a positive integer, say, N. From the indicial equation, $r_1 + r_2 = 1 - q_0^{(1)}$, hence we get

$$r_1 = \frac{1 - q_0^{(1)} + N}{2}. \tag{6.27}$$

The first solution, corresponding to r_1, is given by (6.23). The second solution is of the form

$$y_2(x) = K\, y_1(x) \ln x + Y_2(x), \tag{6.28}$$

where $Y_2(x)$ correct to $O(x^{D+1})$ is given by

$$Y_2(x) = \sum_{n=0}^{D} y_n^* x^{n+r_2} .$$

To find y_n^* explicitly, we substitute (6.28) in (6.22) and get

$$x^2\,Y_2''(x) + xQ_1(x)Y_2'(x) + Q_2(x)Y_2(x) + K\,[2xy_1'(x) + (Q_1(x) - 1)\,y_1(x)] \;=\; 0.$$

Using the series expansion for $Q_1(x)$, $Q_2(x)$ and (6.27), the coefficient of x^r in right hand side becomes zero and we get

$$F(r_2)\,x^{r_2} + \sum_{n\geq 1}\left[F(r_2+n)\,y_n^* + \sum_{j=0}^{n-1}\left\{(r_2+j)\,q_{n-j}^{(1)} + q_{n-j}^{(2)}\right\}\,y_j^*\right]\,x^{r_2+n}$$
$$+K\,y_0^{(1)}\,N\,x^{r_1} + K\sum_{n\geq 1}\left[(2n+N)\,y_n^{(1)} + \sum_{j=1}^{n} q_j^{(1)}\,y_{n-j}^{(1)}\right]\,x^{r_1+n} = 0.$$

Using $r_1 = N + r_2$, and shifting the indices in the last sum, we get

$$\sum_{n\geq 1}\left[F(r_2+n)\,y_n^* + \sum_{j=0}^{n-1}\left\{(r_2+j)\,q_{n-j}^{(1)} + q_{n-j}^{(2)}\right\}\,y_j^*\right]\,x^{r_2+n} + K\,y_0^{(1)}\,N\,x^{r_2+N}$$
$$+K\sum_{n\geq N+1}\left[(2n-N)\,y_{n-N}^{(1)} + \sum_{j=1}^{n-N} q_j^{(1)}\,y_{n-N-j}^{(1)}\right]\,x^{r_2+n} = 0.$$

Breaking the first sum in two parts and recombining the coefficients of x, we get

$$\sum_{r=1}^{N-1}\left[F(r_2+n)\,y_n^* + \sum_{j=0}^{n-1}\left\{(r_2+j)\,q_{n-j}^{(1)} + q_{n-j}^{(2)}\right\}\,y_j^*\right]\,x^{r_2+n}$$
$$+\left[F(r_2+N)\,y_N^* + \sum_{j=0}^{N-1}\left\{(r_2+j)\,q_{N-j}^{(1)} + q_{N-j}^{(2)}\right\}\,y_j^* + K\,y_0^{(1)}\,N\right]\,x^{r_2+N}$$
$$+\sum_{n\geq N+1}\left[F(r_2+n)\,y_n^* + \sum_{j=0}^{n-1}\left\{(r_2+j)\,q_{n-j}^{(1)} + q_{n-j}^{(2)}\right\}\,y_j^*\right.$$
$$\left.+K\left\{(2n-N)\,y_{n-N}^{(1)} + \sum_{j=1}^{n-N} q_j^{(1)}\,y_{n-N-j}^{(1)}\right\}\right]\,x^{r_2+n} = 0.$$

The coefficient of y_N^* is $F(r_2+N) = F(r_1)$, which is zero. Now let the coefficient of each power of x be zero, and that gives us the required K and recurrence relation to find y_n^* explicitly.

Theorem 2: Let the origin be a regular singular point of $Ly = 0$. Rewrite

$Ly = F$ as

$$\sum_{k=0}^{2} x^{2-k} Q_k \partial^{2-k} y(x) = f(x),$$

where

$$\begin{aligned} Q_0(x) &= 1 , \\ Q_k(x) &= x^k \frac{p_{2-k}(x)}{p_2(x)}, \qquad k = 1,2 , \\ f(x) &= \frac{F(x)}{p_2(x)} x^2 . \end{aligned}$$

Since the origin is a regular singular point of $Ly = 0$, $Q_1(x)$ and $Q_2(x)$ are analytic functions at origin. Let

$$Q_k(x) = \sum_{j=0}^{\gamma(k)} q_j^{(k)} x^{j+t_k} ,$$

where t_k is the lowest degree of Q_k, $k = 1,2$.

Let the function $f(x)$ have a series expansion of the form

$$f(x) = \sum_{i=\alpha}^{\beta} f_i x^i .$$

Let $y_1(x)$ and $y_2(x)$ be the first and second Frobenius series solutions respectively, of $Ly = 0$ and r_1 and r_2 be two roots of the indicial equation, such that $r_1 \geq r_2$. Then the form of the particular integral y_I depends on the value of the roots r_1 and r_2 as follows.

<u>$r_1 - r_2 \neq 0$ and not an integer</u>

The particular integral y_I correct to $O(x^{D+1})$ is given by

$$y_I = \sum_{i=\alpha}^{D} y_i x^i . \tag{6.29}$$

$\underline{r_1 = r_2}$

The particular integral y_I correct to $O(x^{D+1})$ is given by

$$y_I = A(\ln x)^2 y_1(x) + A \ln x\, y_1(x) + 2A \ln x\, Y_2(x) + y_3(x) , \qquad (6.30)$$

where

$$\begin{aligned} Y_2(x) &= y_2(x) - \ln x y_1(x) , \\ y_3(x) &= \sum_{i=\alpha}^{D} y_i^{(3)} x^i . \end{aligned}$$

The constant A may or may not be zero, depending on the nonhomogeneous term $f(x)$.

$\underline{r_1 - r_2 = \text{positive integer } N}$

The particular integral y_I of $Ly = f$ correct to $O(x^{D+1})$ is given by

$$y_I = A \ln x\, y_1(x) + B \ln x\, YY(x) + 2B \ln x\, Y_2(x) + y_3(x), \qquad (6.31)$$

where

$$\begin{aligned} y_2(x) &= K y_1(x) \ln x + Y_2(x) , \qquad \text{K is a constant} \\ Y_2(x) &= y_2(x) - K \ln x y_1(x) , \\ YY(x) &= K y_1(x) \ln x , \\ y_3(x) &= \sum_{i=\alpha}^{D} y_i^{(3)} x^i . \end{aligned}$$

The constants A and B may or may not be zero, depending on the function $f(x)$.

The particular integral y_I in all these three cases is unique if

$$\alpha = a - \min_k \{t_k\} , \qquad \beta = D + \min_k \{t_k\} .$$

The coefficient Q_k need only be continued until maximum degree

$$\gamma(k) = D - a + 2\left[\min_k \{t_k\}\right].$$

The solutions $y_1(x)$ and $y_2(x)$ are required only of *Order* D.

Proof: Let $r_1 - r_2 \neq 0$ and not an integer, then substituting (6.29) in $Ly = f$, we get

$$\sum_{m=0}^{2} \sum_{k=0}^{\gamma(k)} \sum_{i=\alpha}^{D} i^{\underline{m}} q_k^{(2-m)} y_i x^{i+k+t_k} = \sum_{i=\alpha}^{\beta} f_i x^i. \tag{6.32}$$

To compare the lowest power of x in both sides, we must have

$$\alpha = a - \min_k \{t_k\}.$$

y_I contains $D - \alpha$ coefficients. To find these coefficients uniquely, we require $D - \alpha$ equations to be solved. Thus we require

$$\begin{aligned} \beta &= D - \alpha + a \\ &= D + \min_k \{t_k\}. \end{aligned}$$

For $m = 0, 1, 2$, the maximum degree of left hand side in (6.32) must be β. Hence the series expansion of the coefficients Q_k need only be continued until maximum degree

$$\gamma(k) = D - 2\alpha + a = D - a + 2\left[\min_k \{t_k\}\right].$$

Similar proof can be applied to the other cases.

Implementation:

When r_1 and r_2 are not integers, then the particular integral y_I can be found simply by comparing the like powers of x in both sides of the differential equation $Ly = f$.

When r_1 and r_2 are integers, y_I may contain a logarithmic term depending on the lower degree a of $f(x)$. Hence to find y_I uniquely, we require the following. When $r_1 = r_2$, we apply the conditions:

- $y^{(3)}[r_1] = 0$,
- if $a > r_1$ then $A = 0$,
- to complete the calculations, D must be greater than r_1.

When $r_1 - r_2$ is a positive integer, then we require that

- $y^{(3)}[r_1] = 0$, $y^{(3)}[r_2] = 0$,
- if $a > r_1$ then both A and B will be zero,
- if $a > r_2$ then $B = 0$, and
- to complete the calculations, D must be greater than r_1.

Theorem 3: Let x_0 be a regular singular or ordinary point of the second order differential equation $Ly = 0$. If the function f does not have Taylor series expansion about x_0, we can obtain a formal expression for a particular integral y_I for the nonhomogeneous equation $Ly = f$ by variation of parameters.

6.8 Analysis of implementations

In this section we analyse various features of the existing implementation of **dsolve** as well as record some of the particular design considerations that help with the efficiency of the package.

6.8.1 Comparison of strategies for ordinary point

The implementation in Maple V Release 3 proceeds as follows. The *ansatz* for y is constructed using D symbolic coefficients y_i. This is substituted into the differential equation, and **series** is used to calculate the series expansion of the left-hand side $L(y)$. The series is collected in x and $D - N$ equations extracted for the y_i. These are given to **solve**, no attempt being made to set them up as a triangular system.

In contrast, the present algorithm solves a triangular system (explicitly), so whereas **solve** requires $(D - N)^3$ operations, the present method uses $(D - N)^2$. In addition, the present algorithm applies the **series** operation separately to each coefficient, whereas Maple V Release 3 applies it to the whole equation, after the ansatz has been substituted. Consequently the present method uses N series calculations on small data objects, whereas Maple V Release 3 does one series calculation on an object at least equal in size to N coefficients plus N derivatives of the ansatz. Finally the present algorithm requires N assignments of $D - N$ rational coefficients to an array, whereas Maple V Release 3 requires N extractions of coefficients in D unknowns.

6.8.2 Treatment of coefficients

In many textbooks, the differential equation is normalized by dividing through by the coefficient of the highest derivative. In this package, however, the user would be advised to present the equation without this normalization, because although one series calculation is avoided, the remaining $N - 1$ become more difficult because they will probably have to be performed on a quotient. Indeed, a future version of

the solver might clear of fractions before starting the expansions.

6.8.3 Variation of parameters

The Wronskian, and the reduced Wronskians, are at present calculated from the fundamental set of solutions. This requires evaluating an N-by-N determinant of series. An alternative would be to calculate the reciprocal of the Wronskian using Abel's relation,

$$1/W = (1/W_0) \exp\left(\int p_{N-1}/p_N dx\right) .$$

The integrand would be obtained by using the **series/divide** routine. For the fundamental set for an ordinary point defined above, $W_0 = 1$. No equivalent to Abel's relation has been published for the reduced Wronskians, and the computation of W_0 for regular singular points has yet to be done.

6.8.4 Comparison of strategies for regular singular points

The strategy for calculating solutions about regular singular points in an Nth degree equation is very expensive computationally because of the symbolic differentiation of series with respect to the index r. Also, in recurrence relation, all the calculations have to be carried out in terms of index r without substituting its numerical value at each step. Therefore, it is important to use the recurrence relation in the form that the expression swell at each step can be avoided.

Maple V Release 3 uses the following recurrence relation to find y_i in the formal solution $\sum_{i\geq 0} y_i x^{i+r}$ (Ince 1957; Della Dorra & Tournier 1981).

Substitute (6.17) in $Ly = 0$ to get

$$\sum_{i=0}^{j} y_i f_{j-i}(r+i) = 0, \qquad j \geq 0 ,$$

where

$$f_j(r) = \sum_{m=0}^{N} r^m q_j^{(N-m)}.$$

The indicial equation, $f_0(r) = 0$, determines N values of r which may or may not be distinct.

y_k can uniquely be determined by

$$y_k = \frac{-y_0}{\prod_{j=1}^{k} f_0(r+j)} F_k(r),$$

where

$$F_k(r) = \begin{vmatrix} f_0(r+1) & 0 & \cdots & 0 & f_1(r) \\ f_1(r+1) & f_0(r+2) & & \cdots & \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ f_{k-2}(r+1) & & & f_0(r+k-1) & \\ f_{k-1}(r+1) & & & f_1(r+k-1) & f_k(r) \end{vmatrix}.$$

$$k = 1, 2, \ldots$$

In this recurrence relation, the determinant has to be computed at each step which increases the computational time.

To find y_k explicitly, our first attempt was to use the recurrence relation (6.18). The coefficients produced by this recurrence relation are continued fractions, and since Maple does not attempt to simplify these unless requested, the differentiation request generates huge amount of work for the system. Thus the coefficients

are simplified at each step before proceeding further. This scheme achieves a remarkable improvement in computational time for some differential equations, but in some cases, the expression for y_k becomes so large that simplifying these expressions at each step through Maple's `simplify` becomes very expensive in terms of memory.

As an alternative, we improve the recurrence relation by recasting it in a form that avoids continued fractions in the coefficients. To do this, we introduce the quantity

$$Y_k = y_k \prod_{i=1}^{k} F(r+i) \tag{6.33}$$

Using the equation (6.18) to obtain an equation for Y_k we find the following.

$$Y_k = -\frac{\prod_{i=1}^{k} F(r+i)}{F(r+k)} \sum_{l=0}^{k-1} \sum_{m=0}^{N} (r+l)^{\underline{m}} q_{k-l}^{(N-m)} y_l ,$$

which gives

$$Y_k = -\prod_{i=1}^{k-1} F(r+i) \sum_{l=0}^{k-1} \sum_{m=0}^{N} (r+l)^{\underline{m}} q_{k-l}^{(N-m)} y_l .$$

Breaking the sum, we get

$$Y_k = -\prod_{i=1}^{k-1} F(r+i) \left[\sum_{l=1}^{k-1} \sum_{m=0}^{N} (r+l)^{\underline{m}} q_{k-l}^{(N-m)} y_l + \sum_{m=0}^{N} r^{\underline{m}} q_k^{(N-m)} y_0 \right] .$$

Substituting y_l from (6.33), we get

$$Y_k = -\prod_{i=1}^{k-1} F(r+i) \left[\sum_{l=1}^{k-1} \sum_{m=0}^{N} (r+l)^{\underline{m}} q_{k-l}^{(N-m)} \frac{Y_l}{\prod_{i=1}^{l} F(r+i)} + \sum_{m=0}^{N} r^{\underline{m}} q_k^{(N-m)} y_0 \right] .$$

Hence the improved version of the recurrence relation is

$$y_k = \frac{Y_k}{\prod_{i=1}^{k} F(r+i)} ,$$

where

$$Y_k = -\sum_{l=1}^{k-1} Y_l \prod_{j=l+1}^{k-1} F(r+j) \sum_{m=0}^{N} (r+l)^{\underline{m}} q_{k-l}^{(N-m)} - \prod_{i=1}^{k-1} F(r+i) \sum_{m=0}^{N} r^{\underline{m}} q_k^{(N-m)} , \qquad k \geq 1 .$$

Use of the modified recurrence relation results in a substantial gain in the computational time and reduction in memory requirement, which is shown in the following section.

6.9 Description of the Package dsolve95

dsolve95 can be used to find the series solutions of Nth order linear nonhomogeneous ordinary differential equations. The package is divided into several procedures to carry out specific tasks. Some procedures are specifically designed for 2nd order differential equations.

Classify_Point: Finds singular points of a Nth order linear ordinary differential equation and classify them as regular or irregular.

Power_Series: Finds power series solutions of a Nth order nonhomogeneous ordinary differential equation about an ordinary point.

Frobenius_Solution_N : Finds Frobenius series solutions to a Nth order linear homogeneous ordinary differential equation about the regular singular point at origin.

Frobenius_Solution_2 : Finds Frobenius series solutions to a 2nd order linear homogeneous ordinary differential equation about the regular singular point at origin.

Pi_Series_N : Finds the particular integral of a Nth order linear nonhomogeneous ordinary differential equation.

Pi_Series_2 : Finds the particular integral of a 2nd order linear nonhomogeneous ordinary differential equation.

Variation_of_Parameters : Finds the particular integral of a Nth order linear nonhomogeneous ordinary differential equation using the method of Variation

of Parameters.

Figure (6.1) explains the order that should be followed to find the general solution to a linear nonhomogeneous ordinary differential equation. A detailed description of all the procedures in `dsolve95` follows.

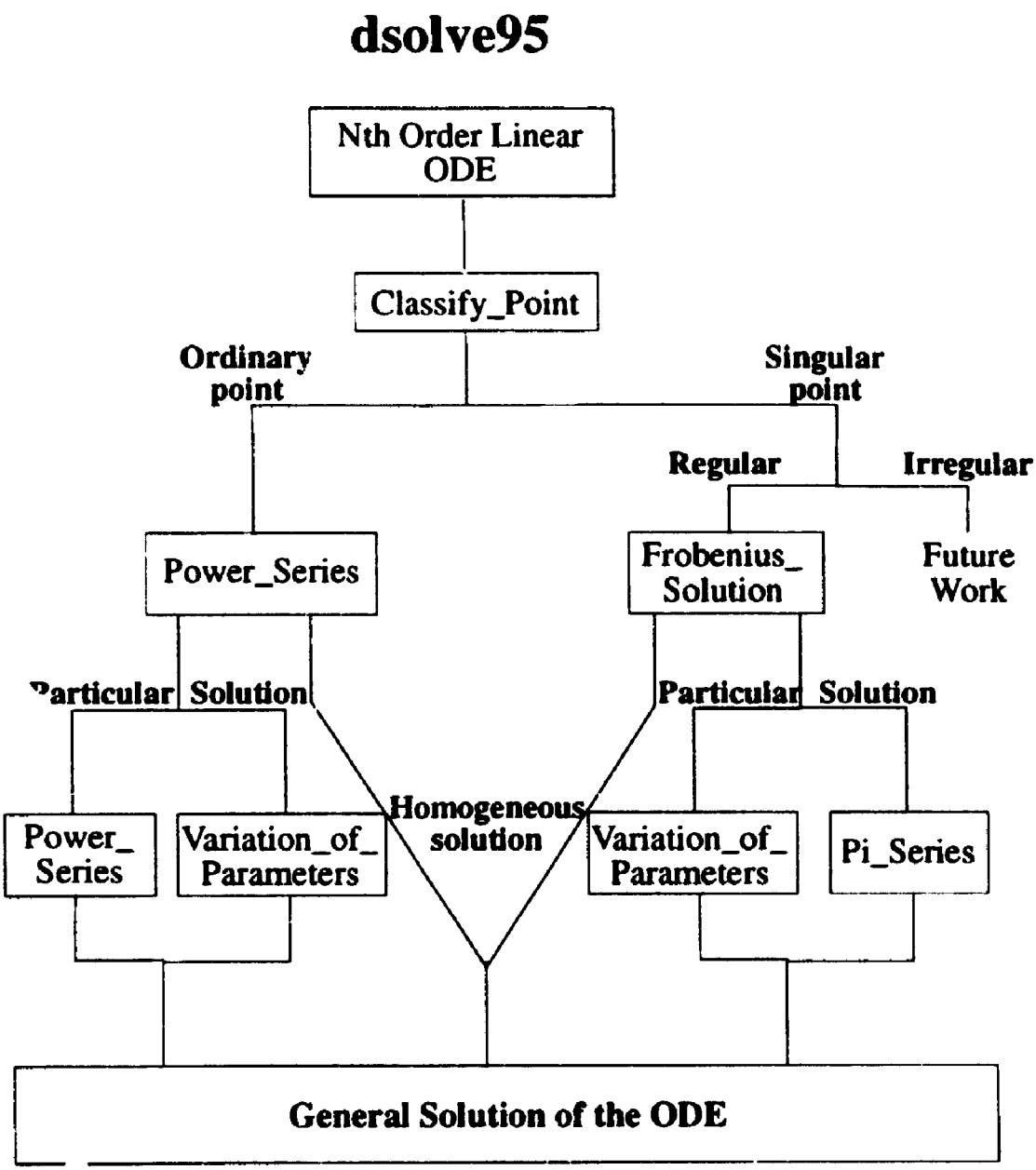

Figure 6.1: Flow diagram for the package `dsolve95`

6.9.1 Classify_Point

Classify the regular and irregular singular points of a Nth order linear ordinary differential equation.

Calling Sequence :

Classify_Point(deqn, var)

Parameters :

deqn — linear ordinary differential equation in var,

var — variable in the differential equation, e.g. $y(x)$.

Synopsis :

- Classify_Point finds all the singular points of a linear Nth order ordinary differential equation and classifies them as regular and irregular singular points.
- If the differential equation does not have any non zero regular singular point, the procedure suggests the user to shift the equation to origin before using the procedure for Frobenius solution, because the procedures for the method of Frobenius in the package can solve the differential equation only about a regular singular point at origin.
- The procedure only provides the information about the nature of the point, and returns null.

Examples :

```
> de:=sin(x)*diff(y(x),x$2)+x*diff(y(x),x)+x*y(x)=0;
```

$$de := \sin(x)\frac{d^2}{dx^2}y(x) + x\frac{d}{dx}y(x) + x\,y(x) = 0$$

```
> Classify_Point(de,y(x));
```

Regular singular points

{3.141592654}

No irregular singular point

None of the regular singular points is zero, therefore rewrite the differential equation in terms of shifted coordinates u = x - x0, x0 being the non-zero regular singular point near which the solution is desired.

```
> de1:=(6*x^2+2*x^3)*diff(y(x),x$2)+21*x*diff(y(x),x)
      +9*(x^2-1)*y(x)=0;
```

$$de1 := (6x^2 + 2x^3)\frac{d^2}{dx^2}y(x) + 21x\frac{d}{dx}y(x) + 9(x^2 - 1)y(x) = 0$$

```
> Classify_Point(de1,y(x));
```

Regular singular points

{0, -3}

No irregular singular point

```
> de2:=x*diff(y(x),x$2)+sin(x)*diff(y(x),x)+x*y(x)=0;
```

$$de2 := x\frac{d^2}{dx^2}y(x) + \sin(x)\frac{d}{dx}y(x) + x\,y(x) = 0$$

```
> Classify_Point(de2,y(x));
```

No singular point

```
> de3:=x^3*(1-x)*diff(y(x),x$2)+(3*x+2)*diff(y(x),x)+x*y(x)=0;
```

$$de3 := x^3(1-x)\frac{d^2}{dx^2}y(x) + (3x+2)\frac{d}{dx}y(x) + x\,y(x) = 0$$

```
> Classify_Point(de3,y(x));
```

Regular singular points

$$\{1\}$$

Irregular singular points

$$\{0\}$$

None of the regular singular points is zero, therefore rewrite the differential equation in terms of shifted coordinates u = x - x0, x0 being the non-zero regular singular point near which the solution is desired.

```
> de4:=x^2*(sin(x-Pi)^2 )*diff(y(x),x$2)+tan(x)*tan(x-Pi)
>       *diff(y(x),x)+(7*x-1)*cos(x)*y(x)=0;
```

$$de4 := x^2(\sin(x-\pi)^2)\frac{d^2}{dx^2}y(x) + \tan(x)\tan(x-\pi)\frac{d}{dx}y(x) + (7x-1)\cos(x)y(x) = 0$$

```
> Classify_Point(de4,y(x));
```

No regular singular point

Irregular singular points

$$\{0\}$$

6.9.2 Power_Series

Power series solutions of a Nth ordinary differential equation.

Calling Sequence :

Power_Series(Deqn, var, pt)

Parameters :

Deqn —linear ordinary differential equation in var or a list of the linear ordinary differential equation with boundary conditions.

var — variable to be solved for.

pt — point about which the solutions are required.

Synopsis :

- The procedure finds the power series solutions of a Nth order linear ordinary differential equation about an ordinary point.
- It expects the differential equation in the form

$$deqn \;:=\; p_N(x)\frac{d^N}{dx^N}y(x) + \cdots + p_1(x)\frac{d}{dx}y(x) + p_0(x)\,y(x) = f(x)$$

- $f(x)$ must have a Taylor series expansion about the given point.
- The point about which the solution is required must be an ordinary point of the given differential equation.
- Derivatives in the boundary conditions should be specified by applying D to the function name, thus the ith derivative of y at the point pt is given as $(D@@i)(y)(pt) = \cdots$.

- The boundary conditions, if supplied, have to be in proper order in form of a list, *e.g.*

  Deqn : $[deqn, y(pt) = \cdots, (D)(y)(pt) = \cdots, (D@@2)(y)(pt) = \cdots, \text{etc.}]$

- If the nonhomogeneous term in the differential equation does not have a Taylor series expansion about the given point, then the procedure returns only the complementary functions. Particular integral in this case can be obtained from the routine **Variation_of_Parameters**.

- If any of the boundary condition is given as type **float**, then the final solution is also given in **float**.

Output :

Returns the power series solutions about the given ordinary point *pt*.

Examples :

```
> de1:=2*diff(y(x),x$2)+x*diff(y(x),x)+y(x)=0;
```

$$de1 := 2\frac{d^2}{dx^2}y(x) + x\frac{d}{dx}y(x) + y(x) = 0$$

```
> Order := 4:
```

```
> Power_Series(de1,y(x),1);
```

$$\_C1(1 - 1/4\,(x-1)^2 + 1/24\,(x-1)^3 + \_C2((x-1) - 1/4\,(x-1)^2 - 1/8\,(x-1)^3)$$

```
> de2 := diff(y(x),x$2)+diff(y(x),x)-x^4*y(x)=sin(2*x);
```

$$de2 := \frac{d^2}{dx^2}y(x) + \frac{d}{dx}y(x) - x^4 y(x) = sin(2x)$$

```
> Order := 8:
```

```
> Power_Series([de2,y(0)=0,D(y)(0)=-2],y(x),0);
```

$$-2x + x^2 - \frac{1}{15}x^5 + \frac{1}{90}x^6 - \frac{3}{70}x^7$$

```
> de3 := diff(y(x),x$2)+x*diff(y(x),x)=x-1;
```

$$de3 := \frac{d^2}{dx^2}y(x) + x\frac{d}{dx}y(x) = x - 1$$

```
> Order := 6:
```

```
> Power_Series([de3,y(2)=1.0,D(y)(2)=-4.0],y(x),2);
```

$$9.0 - 4.0x + 4.500000000(x-2)^2 - 2.166666667(x-2)^3$$

$$+.3333333336(x-2)^4 + .1916666666(x-2)^5$$

6.9.3 Frobenius_Solution_N

Frobenius series solutions of a Nth order linear ordinary differential equation about a regular singular point at origin.

Calling Sequence :

Frobenius_Solution_N(deqn, var)

Parameters :

deqn —Nth order linear ordinary differential equation in var.

var — variable to be solved for.

Synopsis :

- The procedure finds the Frobenius series solutions of a Nth order linear ordinary differential equation about a regular singular point at origin.
- It expects the differential equation in the form

$$deqn \;:=\; p_N(x)\,\frac{d^N}{dx^N}y(x) + \cdots + p_1(x)\,\frac{d}{dx}y(x) + p_0(x)\,y(x) = 0\ .$$

- If the regular singular point is not at origin but at $x = x_0$, then the differential equation should be written in terms of shifted coordinates $u = x - x_0$, before calling the routine.
- For a 2nd order ordinary differential equation, the routine `Frobenius_Solution_2` should be called, which is much faster and efficient.

Output :

Returns the Frobenius series solutions about the regular singular point at origin.

Example :

```
> de1:=x^2*diff(y(x),x$3)+3*x*diff(y(x),x$2)
                          +diff(y(x),x)-y(x)=0;
```

$$de1 := x^2 \frac{d^3}{dx^3}y(x) + 3x \frac{d^2}{dx^2}y(x) + \frac{d}{dx}y(x) - y(x) = 0$$

```
> Frobenius_Solution_N(de1,y(x));
```

$$\_C1\%1 + \_C2\left(\%1 \ln(x) - \frac{11}{432}x^3 - \frac{25}{55296}x^4 - 3x - \frac{9}{16}x^2 - \frac{137}{3456000}x^5\right)$$
$$+\_C3\left(\%1 \ln(x)^2 + (\frac{-9}{8}x^2 - \frac{11}{216}x^3 - \frac{25}{27648}x^4 - \frac{137}{1728000}x^5 - 6x)\ln(x)\right.$$
$$\left. +12x + 3x^2 + \frac{103}{648}x^3 + \frac{65}{20736}x^4 + \frac{7697}{259200000}x^5\right)$$
$$\%1 := 1 + x + 1/8x^2 + 1/216x^3 + 1/13824x^4 + 1/1728000x^5$$

6.9.4 Frobenius_Solution_2

Frobenius series solutions of a 2nd order linear ordinary differential equation about a regular singular point at origin.

Calling Sequence :

Frobenius_Solution_2(deqn, var)

Frobenius_Solution_2(deqn, var, r)

Parameters :

deqn —a 2nd order linear ordinary differential equation in var.

var — variable to be solved for.

r — (optional) returns the list of the roots of the indicial equation .

Synopsis :

- The procedure finds the Frobenius series solutions of a 2nd order linear ordinary differential equation about a regular singular point at origin.
- It expects the ordinary differential equation in the form

$$deqn \;:=\; p_2(x)\frac{d^2}{dx^2}y(x) + p_1(x)\frac{d}{dx}y(x) + p_0(x)\,y(x) = 0\,.$$

- If the regular singular point is not at origin but at $x = x_0$, then the differential equation should be written in terms of shifted coordinates $u = x - x_0$, before calling the routine.
- The optional parameter 'r', returns the list of the roots of the indicial equation $[r_1, r_2]$.

Output :

Returns the Frobenius series solutions of a 2nd order linear ordinary differential equation about the regular singular point at origin and the roots of the indicial equation (optional).

Examples :

```
> de1:=x^2*diff(y(x),x$2)+x^2*diff(y(x),x)-2*y(x)=0;
```

$$de1 := x^2 \frac{d^2}{dx^2} y(x) + x^2 \frac{d}{dx} y(x) - 2\,y(x) = 0$$

```
> Order := 7:
```

```
> Frobenius_Solution_2(de1,y(x));
```

$$\_C1(x^2 - \frac{1}{2}x^3 + \frac{3}{20}x^4 - \frac{1}{30}x^5 + \frac{1}{168}x^6) + \_C2\frac{(1 - 1/2\,x)}{x}$$

```
> de2:=x^2*diff(y(x),x$2)+x*diff(y(x),x)+x^2*y(x)=0;
```

$$de2 := x^2 \frac{d^2}{dx^2} y(x) + x \frac{d}{dx} y(x) + x^2\,y(x) = 0$$

```
> Frobenius_Solution_2(de2,y(x),r);
```

$$\_C1(1 - \frac{1}{4}x^2 + \frac{1}{64}x^4) + \_C2\left((1 - \frac{1}{4}x^2 + \frac{1}{64}x^4)\ln(x) + \frac{1}{4}x^2 - \frac{3}{128}x^4\right)$$

```
> r;
```

$$[0, 0] \qquad (6.34)$$

6.9.5 Pi_Series_N

Particular integral of a Nth order linear nonhomogeneous ordinary differential equation with a regular singular point at origin.

Calling Sequence :

Pi_Series_N(deqn, var)

Parameters :

deqn —a Nth order linear nonhomogeneous ordinary differential equation with regular singular point at origin.
var — variable in the given diferential equation, e.g. $y(x)$.

Synopsis :

- The procedure finds the particular integral of a Nth order linear nonhomogeneous ordinary differential equation.
- It expects the ordinary differential equation in the form

$$deqn \;:=\; p_N(x)\frac{d^N}{dx^N}y(x) + \cdots + p_1(x)\frac{d}{dx}y(x) + p_0(x)\,y(x) = f(x)\;.$$

- $f(x)$ must have a Taylor series exansion.

Output :

Returns the particular integral uniquely of a Nth order linear ordinary differential equation.

Examples :

```
> de1:= x^2*diff(y(x),x$2)+3*x*diff(y(x),x)+5*y(x)=x^2;
```

$$de1 := x^2\frac{d^2}{dx^2}y(x) + 3x\frac{d}{dx}y(x) + 5\,y(x) \;=\; x^2$$

```
> Pi_Series_N(de1,y(x));
```

$$1/13\,x^2$$

```
> de2:=x^2*diff(y(x),x$2)+3*x*diff(y(x),x)
>       -(x^2+x*coth(x))*y(x)=-2*x;
```

$$de2 := x^2 \frac{d^2}{dx^2}y(x) + 3 * x \frac{d}{dx}y(x) - (x^2 + x\coth(x))\,y(x) \;=\; -2x$$

```
> Order:=10:
> Pi_Series_N(de2,y(x));
```

$$-x - 2/21\,x^3 - \frac{11}{3570}\,x^5 - \frac{11}{166005}\,x^7 - \frac{41}{409965948}\,x^9$$

```
> de3:=x^3*diff(y(x),x$3)+x^2*diff(y(x),x$2)+x*diff(y(x),x)
> +x^2*y(x)=x^4;
```

$$de3 := x^3\frac{d^3}{dx^3}y(x) + x^2\frac{d^2}{dx^2}y(x) + x\frac{d}{dx}y(x) + x^2\,y(x) \;=\; x^4$$

```
> Order := 10:
> Pi_Series_N(de3,y(x));
```

$$1/40\,x^4 - 1/6240\,x^6 + 1/2496000\,x^8 - 1/2046720000\,x^{10}$$

6.9.6 Pi_Series_2

Particular solution of a 2nd order linear nonhomogeneous ordinary differential equation with regular singular point at origin.

Calling Sequence :

Pi_Series_2(deqn, var, r, sol)

Parameters :

deqn —a 2nd order nonhomogeneous linear ordinary differential equation with regular singular point at origin.

var — variable in 'deqn'.

r — list of the roots of the indicial equation.

sol — Frobenius series solutions of the homogeneous 'deqn'.

Synopsis :

- The procedure finds the particular integral of a 2nd order linear nonhomogeneous ordinary differential equation with regular singular point at origin.
- It expects the ordinary differential equation in the form

$$deqn \;:=\; p_2(x)\frac{d^2}{dx^2}y(x) + p_1(x)\frac{d}{dx}y(x) + p_0(x)\,y(x) = f(x)\;.$$

- $f(x)$ must have a Taylor series exansion.
- The roots of the indicial equation should be in the form of a list, e.g. $[r_1, r_2]$. It can be obtained as an optional parameter from the procedure **Frobenius_Solution_2**.

- The fourth parameter, sol, is the Frobenius series solution of the differential equation, which can be obtained from the procedure `Frobenius_Solution_2`.

Output :

Returns the particular integral uniquely of a 2nd order nonhomogeneous linear ordinary differential equation about a regular singular point at origin.

Examples :

```
> Order:=6:
> de1:=x^2*diff(y(x),x$2)+3*x*diff(y(x),x)-(x^2+x*coth(x))*y(x)
>       =-2*x*exp(-x);
```

$$de1 := x^2 \frac{d^2}{dx^2}y(x) + 3x \frac{d}{dx}y(x) - (x^2 + x\coth(x))\, y(x) = -2x\exp(-x)$$

```
> F:= Frobenius_Solution_2(de1,y(x),r):
> r;
```

$$[-1 + \sqrt{2}, -1 - \sqrt{2}]$$

```
> P:=Pi_Series_2(de1,y(x),r,F);
```

$$P := -x + 2/7x^2 - 1/6x^3 + 5/161x^4 - 1/120x^5$$

```
> Order := 6:
> de2:=x^2*diff(y(x),x$2)-3*x*diff(y(x),x)+(4*x+4)*y(x)
>       = x+x^2+x^(-3)+x^4+(1/x);
```

$$de2 := x^2 \frac{d^2}{dx^2}y(x) - 3x \frac{d}{dx}y(x) + (4x + 4)\, y(x) = x + x^2 + x^{-3} + x^4 + \frac{1}{x}$$

```
> F:= Frobenius_Solution_2(de2,y(x),r):
```

```
> r,
```

$$[2, 2]$$

```
> P:=Pi_Series_2(de2,y(x),r,F);
```

$$P := \frac{1}{25x^3} - \frac{1}{100x^2} + \frac{26}{225x} - \frac{26}{225} + \frac{329}{225}x + \left(-\frac{1091}{450}\ln(x) - \frac{1091}{450}\ln(x)^2\right)x^2 + \left(-\frac{2182}{75}\ln(x) + \frac{8728}{225} + \frac{2182}{225}\ln(x)^2\right)x^3 + \left(-\frac{2182}{225}\ln(x)^2 + \frac{2182}{45}\ln(x) - \frac{73963}{900}\right)x^4$$

```
> Order := 6:
```

```
> de3:=x^2*diff(_  ),x$2)+x^2*diff(y(x),x)-2*y(x)
```

```
>       = x+x^2+x^(-2)+x^(-3)+x^4+(1/x);
```

$$de3 := x^2\frac{d^2}{dx^2}y(x) + x^2\frac{d}{dx}y(x) - 2\,y(x) = x + x^2 + x^{-2} + x^{-3} + x^4 + \frac{1}{x}$$

```
> F:= Frobenius_Solution_2(de3,y(x),r):
```

```
> r;
```

$$[2, -1]$$

```
> P:=Pi_Series_2(de3,y(x),r,F);
```

$$P := \frac{1}{10x^3} + \frac{13}{40x^2} - \frac{11}{20}\frac{\ln(x)}{x} + \frac{11}{40}\ln(x) - \frac{33}{80} - \frac{29}{80}x + \frac{109}{280}\ln(x)x^2 + \left(\frac{109}{640} - \frac{109}{480}\ln(x)\right)x^3 + \left(\frac{2297}{96000} + \frac{109}{1600}\ln(x)\right)x^4$$

```
> Order:=9:
```

```
> de4:=x^2*diff(y(x),x$2)-12*y(x) = x^4;
```

$$de4 := x^2\frac{d^2}{dx^2}y(x) - 12\,y(x) = x^4$$

```
> F:= Frobenius_Solution_2(de4,y(x),r):
```

```
> r;
```

$$[4, -3]$$

```
> P:=Pi_Series_2(de1,y(x),r,F);
```

$$P := \frac{1}{7}\ln(x)\,x^4$$

6.9.7 Variation_of_Parameters

Particular solution of a Nth order nonhomogeneous linear ordinary differential equation using the method of Variation of parameters.

Calling Sequence :

Variation_of_Parameters(sol, g, ord, indepvar)

Parameters :

sol —set of N linearly independent solutions of the Nth order linear homogeneous ordinary differential equation.

g — $\frac{\text{nonhomogeneous term}}{\text{coefficient of the highest derivative}}$

ord — order of the ordinary differential equation.

indep_var — Independent variable.

Synopsis :

- The procedure finds the particular integral of a Nth order linear nonhomogeneous ordinary differential equation using the method of Variation of parameters.
- It can be used if the nonhomogeneous term has a Taylor series exansion or not.
- Prior to calling this procedure, the solution set to the homogeneous ordinary differential equation must be obtained using an appropriate procedure.

Output :

Returns the particular integral of a Nth order linear nonhomogeneous ordinary differential equation.

Examples :

```
> de1:=x^2*diff(y(x),x$2)+x*diff(y(x),x)-4*y(x)=ln(x);
```

$$de1 := x^2 \frac{d^2}{dx^2}y(x) + x \frac{d}{dx}y(x) - 4\,y(x) = \ln x$$

```
> Frobenius_Solution_2(de1,y(x)):
> sol[1]:= subs(_C1=0,_C2=1,"):
> sol[2]:= subs(_C1=1,_C2=0,""):
> g:= ln(x)/x^2:
> Variation_of_Parameters(sol,g,2,x);
```

$$1/4\ \ln(1/x)$$

```
> de2:=3*x^2*diff(y(x),x$2)+11*x*diff(y(x),x)-3*y(x)=8-3*ln(x);
```

$$de2 := 3x^2 \frac{d^2}{dx^2}y(x) + 11 * x \frac{d}{dx}y(x) - 3\,y(x) = 8 - 3\ln(x)$$

```
> Frobenius_Solution_2(de2,y(x)):
> sol[1]:= subs(_C1=0,_C2=1,"):
> sol[2]:= subs(_C1=1,_C2=0,""):
> g:= (8-3*ln(x))/(3*x^2):
> Variation_of_Parameters(sol,g,2,x);
```

$$-\ln(1/x)$$

```
> de3:=x^2*diff(y(x),x$2)+x^2*diff(y(x),x)-2*y(x)=x+x^2+(1/x);
```

$$de3 := x^2 \frac{d^2}{dx^2}y(x) + x^2 \frac{d}{dx}y(x) - 2\,y(x) = x + x^2 + \frac{1}{x}$$

```
> Order := 5:
> Frobenius_Solution_2(de3,y(x)):
> sol[1]:= subs(_C1=0,_C2=1,''):
> sol[2]:= subs(_C1=1,_C2=0,''):
> g:=(x+x^2+(1/x))/x^2:
> V:= Variation_of_Parameters(sol,g,2,x);
```

$$
\begin{aligned}
V \;:=\; & \left(-\frac{1}{9}-\frac{1}{3}\ln(x)\right)\frac{1}{x}+\left(\frac{1}{6}\ln(x)-\frac{7}{36}\right)-\frac{5}{12}x \\
& +\left(\frac{17}{36}\ln(x)+\frac{1}{12}\right)x^2+\left(-\frac{17}{72}\ln(x)+\frac{13}{96}\right)x^3 \\
& +\left(-\frac{959}{14400}+\frac{17}{240}\ln(x)\right)x^4
\end{aligned}
$$

6.10 Comparison of dsolve95 and dsolve/series

Several ordinary differential equations are solved using **dsolve95** and **dsolve/series** routine in Maple and their results are compared. Several aspects, in which **dsolve95** achieved improvement over **dsolve/series** are as follows.

6.10.1 Correct Homogeneous Solution

dsolve95 was used to examine several ordinary differential equations, for which **dsolve/series** provides incorrect homogeneous solution [section (6.2.1)]. Our routine gives correct homogeneous solution for these ordinary differential equations. For example, solutions of (6.1) (Latta & Hess 1973) and eqn.(78) from O'neill & Stewartson (1967) obtained by **dsolve95** match with those given by the authors.

6.10.2 Particular Integral

As discussed in the sections (6.2.2) and (6.3.1), **dsolve/series** does not provide the particular integral to the nonhomogeneous ordinary differential equation with a regular singular point. Also, in case of the series solution about an ordinary point, **dsolve/series** fails if the nonhomogeneous term in the differential equation does not have a Taylor series expansion about that point.

The routine **dsolve95** removes these errors/shortcomings from existing package in Maple V Release 3 and finds the particular integral for nonhomogeneous ordinary differential equation.

- If x_0 is a regular singular point of the nonhomogeneous differential equation $Ly = f(x)$, then **dsolve95** finds the particular solution irrespective of whether the nonhomogeneous term has a Taylor series expansion or not.
  - If the nonhomogeneous term in the differential equation has a Taylor series expansion about that point, then **dsolve95** uses the routine **Pi_Series_N** for Nth order and routine **Pi_Series_2** for 2nd order ordinary differential equations and finds the particular integral which is computationaly unique.
  - Otherwise, it uses the routine **Variation_of_Parameters**.
- If the solution is required about an ordinary point of the nonhomogeneous differential equation $Ly = f(x)$, again **dsolve95** finds the particular solution irrespective of whether the nonhomogeneous term has a Taylor series expansion or not.

- If the nonhomogeneous term in the differential equation has a Taylor series expansion about that point, then **dsolve95** uses the routine **Power_Series**.
- Otherwise, the routine **Variation_of_Parameters** is used.

6.10.3 Real Solutions derived from the Complex solutions

Let us suppose that the indicial equation of an ordinary differential equation, with regular singular point at origin, has the root $r = a+ib$. The trivial solution formally is,

$$y(x) = x^{a+ib} \sum_{m\geq 0} y_m x^m, \qquad x > 0 \,. \tag{6.35}$$

Rewrite

$$x^{a+ib} \;=\; x^a \left\{\cos(b\ln x) + i\sin(b\ln x)\right\}.$$

Since the coefficients y_m in (6.35) are complex, we write $y_m = y_m^{(r)} + iy_m^{(i)}$. Therefore,

$$\begin{aligned} y(x) \;=\; x^a \Bigg[& \left(\cos(b\ln x) \sum_{m\geq 0} y_m^{(r)} x^m - \sin(b\ln x) \sum_{m\geq 0} y_m^{(i)} x^m \right) \\ & + i \left(\cos(b\ln x) \sum_{m\geq 0} y_m^{(i)} x^m + \sin(b\ln x) \sum_{m\geq 0} y_m^{(r)} x^m \right) \Bigg] . \end{aligned}$$

Equivalently,

$$y(x) = Y1(x) + iY2(x) \,. \tag{6.36}$$

Lemma: Let $y(x) = Y1(x) + i\,Y2(x)$ be a complex-valued solution of the differential equation, then $Y1(x)$ and $Y2(x)$ are its real-valued linearly independent solutions. Hence from (6.36), the solution of the ordinary differential equation is

$$y(x) \;=\; \_C1\,Y1(x) + \_C2\,Y2(x) \,.$$

dsolve95 gives real valued solutions, as already exemplified in section(6.3.3).

6.10.4 Solution in the factored form

For some differential equations, the roots of the indicial equation are such that the expressions in the solution becomes larger as the number of terms increases. **dsolve95** provides the solution in a factored form which is compact. For example, the solutions of the equation

$$x^2\frac{d^2y}{dx^2}+(x^2-3x)\frac{dy}{dx}+(x-4)y \;=\; 0$$

by **dsolve/series** are given as

$$\begin{aligned}
&_C1x^{(2-2\sqrt{2})}\Big(1-\frac{3-2\sqrt{2}}{-11+4\sqrt{2}+(2-2\sqrt{2})^2}x\\
&\qquad+\frac{(4-2\sqrt{2})(3-2\sqrt{2})}{(-11+4\sqrt{2}+(2-2\sqrt{2})^2)(-8+(2-2\sqrt{2})^2)}x^2\\
&\quad-\frac{(5-2\sqrt{2})(4-2\sqrt{2})(3-2\sqrt{2})}{(-11+4\sqrt{2}+(2-2\sqrt{2})^2)(-8+(2-2\sqrt{2})^2)(-3-4\sqrt{2}+(2-2\sqrt{2})^2)}x^3\\
&\qquad+O(x^4)\Big)\\
&+_C2x^{(2+2\sqrt{2})}\Big(1-\frac{3+2\sqrt{2}}{-11-4\sqrt{2}+(2+2\sqrt{2})^2}x\\
&\qquad+\frac{(4+2\sqrt{2})(3+2\sqrt{2})}{(-11-4\sqrt{2}+(2+2\sqrt{2})^2)(-8+(2+2\sqrt{2})^2)}x^2\\
&\quad-\frac{(5+2\sqrt{2})(4+2\sqrt{2})(3+2\sqrt{2})}{(-11-4\sqrt{2}+(2+2\sqrt{2})^2)(-8+(2+2\sqrt{2})^2)(-3+4\sqrt{2}+(2+2\sqrt{2})^2)}x^3\\
&\qquad+O(x^4)\Big).
\end{aligned}$$

Present package gives the solution in the factored form as

$$\begin{aligned}
&_C1x^{(2+2\sqrt{2})}\Big(1+(-\frac{13}{31}-\frac{10}{31}\sqrt{2})x+(\frac{43}{217}+\frac{59}{434}\sqrt{2})x^2\\
&\qquad+(-\frac{869}{14973}-\frac{421}{9982}\sqrt{2})x^3\Big)
\end{aligned}$$

$$+\_C2x^{(2-2\sqrt{2})}\left(1+(-\frac{13}{31}+\frac{10}{31}\sqrt{2})x+(\frac{43}{217}-\frac{59}{434}\sqrt{2})x^2\right.$$
$$\left.+(-\frac{869}{14973}+\frac{421}{9982}\sqrt{2})x^3\right).$$

Preference for a particular form of the solution depends on the user.

6.10.5 Solution of the Indicial Equation

dsolve/series uses the Maple V Release 3 routine **solve** to get the roots of the indicial equation. When the indicial equation of the given differential equation is an irreducible polynomial or cannot be factorized completely, **solve** does not provide all the roots explicitly. When **solve** is unable to find any solution, the empty sequence NULL is returned. For example, **solve** gives the roots of the indicial equation

$$\texttt{Indic}: r^7 - 4r^4 + r^3 - 2r + 5 \tag{6.37}$$

as

$$\text{RootOf}(\_Z^7 - 4\_Z^4 + \_Z^3 - 2\_Z + 5).$$

In such cases, Maple V Release 3 code stops. Instead of using only **solve**, **dsolve95** uses the following steps,

```
> solve(Indic,r):
> if has({"},RootOf) or {"} = {} then
>       Solve95([solve(Indic,r)]):
> fi:
```

The routine **Solve95** finds all the roots using strategy given in previous chapter.

6.10.6 Roots of the Indicial equation differ by an integer

As discussed before, when the roots of the indicial equation differ by an integer, the Frobenius series solutions do not always contain logarithmic term. **dsolve95** recognizes this fact and presents the solution in proper form. Consider the differential equation,

$$x^2 y'' - 4x y' + 4y = 0.$$

```
> x^2*diff(y(x),x$2) -4*x*(diff(y(x),x) + 4*y(x) = 0:
> Frobenius_Solution_2('',y(x));
```

$$y(x) = \_C1\, x^4 + \_C2\, x$$

6.10.7 Consistent Interface

dsolve95 uses same notations for the arbitrary constants in the solution in both cases, the method of Frobenius and the power series method about an ordinary point. For example, differential equation **de1** is solved about an ordinary point, $x = 0$, while **de2** is solved about the regular singular point at origin.

```
> de1:= diff(y(x),x$2) +sin(x)*diff(y(x),x) + x*y(x) = x:
> Order:= 6:
> Power_Series(de1, y(x), 0);
```

$$y(x) = \_C1(1 - 1/6\,x^3 + 1/40\,x^5) + \_C2(x - 1/6\,x^3 - 1/12\,x^4 + 1/30\,x^5) + 1/6x^3 - 1/40\,x^5$$

```
> de2:= x^2*diff(y(x),x$2)-2*x*diff(y(x),x)-(x^2-2)*y(x)=0:
> Frobenius_Solution_2(de2, y(x));
```

$$y(x) = \_C1\, x^2(1 + 1/6x^2) + \_C2\, x(1 + 1/2x^2)$$

6.10.8 Power series solution about a non-zero ordinary point

If the boundary conditions are not prescribed, Maple V Release 3 routine does not have an option to find the solutions of a differential equation about a non-zero ordinary point. The **args[3]** in the routine **Power_Series** is the point about which the solution is required, therefore the solutions can be obtained about any ordinary point irrespective of whether the boundary conditions are given or not. For example,

```
> Order := 4:
> 2*diff(y(x),x$2)+x*diff(y(x),x)+y(x)=0:
> Power_Series(",y(x),1);
```

$$\_C1(1 - 1/4\,(x-1)^2 + 1/24\,(x-1)^3 + \_C2((x-1) - 1/4\,(x-1)^2 - 1/8\,(x-1)^3)$$

6.10.9 Efficiency and CPU Time

In terms of the memory requirement and the computational time taken, **dsolve95** is a major improvement over the existing routine **dsolve/series** in Maple V Release 3. In the computer algebra systems, these improvements are of utmost importance. These improvements are achieved because of the use of recurrence relations in the routines and other simplifications at intermediate stages of algorithms.

For any Nth order linear ordinary differential equation with regular singular point at origin, **dsolve95** uses the routine **Frobenius_Solution_N** based on the method given in section (6.6.5). Routine **Frobenius_Solution_2** is a specific routine only for second order differential equations with regular singular point at origin. In all three cases, when the roots of the indicial equation are different, equal or differ by an integer, this routine finds Frobenius series solutions explicitly by using the recurrence relations given in section (6.7). CPU time and the words allocation required by these routines in all of the three cases are compared with those required by Maple V Release 3 routine **dsolve** in Tables (6.1), (6.2) and (6.3).

Case 1: Consider the linear ordinary differential equation (O'neill & Stewartson 1967).

$$x^3K'y'' + x(x^2K'' + 3xK' + 2K')y' - (x^2K'' + 4xK' + 2K)y = 0,$$

where

$$K = \frac{1}{x} - \coth x.$$

The roots of its indicial equation are $[-2 + \sqrt{10}, -2 - \sqrt{10}]$.

Table (6.1) compares the CPU time and the memory requirements needed by the routines **dsolve/series** and **dsolve95** for solving this equation for different number of terms in the series solution.

Case 2: Consider the differential equation

$$K_1 y'' + K_2 y' + K_3 y = 0,$$

where

$$K_1 = x\sinh x + ex\cosh x\ .$$

Order	CPU time (seconds)			Words		
	Maple's	dsolve95		Maple's	dsolve95	
	dsolve	Nth Order	2nd Order	dsolve	Nth Order	2nd Order
25	1298	61	13	720,764	507,811	376,673
30	5228	116	16	1,048,384	704,383	393,144
35	19194	211	19	1,326,861	884,574	393,144
45	159565	722	28	2,391,626	1,343,242	409,525
75			93			442,287
100			257			933,717

Table 6.1: Comparison of CPU time and memory requirements for the method of Frobenius, when the roots of the indicial equation are different.

$$K_2 = 2x\cosh x + \sinh x + 2ex\sinh x + e\cosh x ,$$

$$K_3 = e^x ,$$

$$e = 0.01 .$$

The roots of the indicial equation are $[0, 0]$.

The computational time and memory requirements for the routines are compared in Table (6.2).

Case 3: The roots of the indicial equation of the differential equation

$$25x(1-x^2)y'' - 20(5x-2)y' + (25x - \frac{4}{x})y = 0$$

are $[\frac{1}{5}, -\frac{4}{5}]$ (an integer difference). Table (6.3) compares the CPU time and words

Order	CPU time (seconds)			Words		
	Maple's	dsolve95		Maple's	dsolve95	
	dsolve	Nth Order	2nd Order	dsolve	Nth Order	2nd Order
20	100	22	1	491,430	409,525	180,191
30	481	84	2	786,288	491,430	229,334
40	1526	238	4	1,081,146	655,240	344,001
50	4053	554	6	2,096,768	868,193	360,382
80			18			671,621
100			35			1,146,670

Table 6.2: Comparison of CPU time and memory requirements for the method of Frobenius, when the roots of the indicial equation are equal.

allocation required by different routines to solve this equation for different number of terms in the series solution.

For the power series solution about an ordinary point, **dsolve/series** uses the direct series method and solves the systems of equations. However, **dsolve95** uses recurrence relations (6.15) or (6.16).

The use of recurrence relations and other significant changes in the programming structure give a remarkable achievement in CPU time and the memory requirements. Table (6.4) compares the CPU time and the memory requirements needed by the routines **dsolve/series** and **dsolve95** for solving the following linear

Order	CPU time (seconds)			Words		
	Maple's	dsolve95		Maple's	dsolve95	
	dsolve	Nth Order	2nd Order	dsolve	Nth Order	2nd Order
20	738	33	3	655,240	327,620	147,429
30	14483	207	4	2,145,911	688,002	163,810
40		886	5		1,408,766	229,334
50		2778	9		2,604,579	229,334
75			40			278,477
100			123			327,620

Table 6.3: Comparison of CPU time and memory requirements for the method of Frobenius, when the roots of the indicial equation differ by an integer.

ordinary differential equation about an ordinary point at origin (without boundary conditions), for different number of terms in the solution.

$$y'' + \cos x y' + 4y = 2x - 1 .$$

In the case when the boundary conditions are given in type **float** for a linear ordinary differential equation about an ordinary point, **dsolve/series** is fast, but is not memory efficient. For example, consider the equation,

$$x^3 K' y'' + x(x^2 K'' + 3x K' + 2K')y' - (x^2 K'' + 4x K' + 2K)y = -x^2 K1'' ,$$

where

$$K = \frac{1}{x} - \coth x \qquad \text{and} \qquad K1 = \coth x - 1 .$$

Order	CPU time (seconds)		Words	
	dsolve	dsolve95	dsolve	dsolve95
20	22.92	0.01	360,382	16,381
30	86.12	2.30	393,144	212,953
40	233.82	4.81	425,906	245,715
50	519.06	7.64	458,668	245,715
60	1020.30	12.74	491,430	262,096
70	1819.96	18.27	540,573	262,096

Table 6.4: Comparison of CPU time and memory requirements for the power series solution of a linear ordinary differential equation about an ordinary point without boundary conditions.

The boundary conditions are

$$y(5/2) = 1.2345678, \qquad D(y)(5/2) = 8.7654321.$$

Comparison with **dsolve** in Table (6.5) shows that the routine **Power_Series** in the present package is very fast and memory efficient.

Remark:

- For the Tables (6.1), (6.2) and (6.3), the routines were run on a 486 DX-4, 100 MHz system.
- For the Tables (6.4) and (6.5), the routines were run on a IBM RISC-6000 system.

Order	CPU time (seconds)		Words	
	dsolve	dsolve95	dsolve	dsolve95
10	6.82	0.01	933,717	16,381
15	28.25	1.16	3,587,439	212,953
18	60.16	2.28	7,158,497	278,477
20	90.13	2.55	10,795,079	294,858
25	252.02	4.39	26,422,553	311,239
40		11.15		393,144
60		21.47		786,288
100		67.73		1,457,909

Table 6.5: Comparison of CPU time and memory requirements for the power series solution of a linear ordinary differential equation about an ordinary point with floating point boundary conditions.

Chapter 7

Summary and Conclusions

> *This is not the end. It is not even the beginning of the end. But it is, perhaps, the end of the beginning.*
>
> Winston Churchill, *Of the Battle of Egypt.*

Several routines, to be used in computer algebra systems, are developed in this thesis to solve the linear nonhomogeneous ordinary differential equations with an ordinary or regular singular point. These routines show significant improvements over existing routines in Maple V Release 3 in terms of memory and computational time requirements. The code for **dsolve/series** in Maple V Release 3 was written by an undergraduate student using a naive method proposed in standard text books. **dsolve95** provides the correct answer for many differential equations for which **dsolve/series** gives the wrong answer. In addition, **dsolve95** determines series solutions much more efficiently. Using these tools, a large number of terms in the series expansions can be included to get highly accurate solutions of ordinary differential equations.

An algorithm to obtain the convenient solution of polynomial equations in

terms of a set of algebraic extensions dependent over the rationals is developed and implemented in Maple.

Tools for managing the problem of intermediate expression swell in computer algebra systems have also been developed. Suggestions for further improvements, to be included in future versions of Maple, are also made.

Using these tools, highly accurate solutions of differential equations are obtained to calculate the couple acting on a spherical particle touching a plane wall. The traditionally used numerical solutions in such cases are replaced with a large number of terms in the series solution. The series solution is then expanded along the real axis and matched to the asymptotic solution. This provides a more accurate solution than a numerical one with less effort.

The difficulties in the numerical solution prevented O'Neill & Stewartson (1967) from solving the differential equation to reasonable accuracy for calculating couples acting on a sphere and the couple acting on the plane wall. They were unable to decide whether the two constants occurring in the solution are equal or not. With the increased accuracy in the series solutions, we found conclusively that the two constants are indeed equal to 10 significant digits.

The problem of inviscid irrotational flow past a sphere touching a plane wall is also solved (previously dealt by Latta & Hess 1973) accurately using a large number of terms in the series expansion.

APPENDIX A
Computer Programs for Chapter 3

```
#-------------------------------------------------------------
#  THE PROGRAM IS DIVIDED IN 2 PARTS:
#  1. FIND  SERIES SOLUTIONS ABOUT EXPANSIONS POINTS AND
#     ASYMPTOTIC SOLUTION FOR s --> INFINITY
#  2. FIND THE CONSTANTS K AND k
#-------------------------------------------------------------

#  1. FIND  SERIES SOLUTIONS ABOUT EXPANSIONS POINTS

# READ PROCEDURE FOR THE METHOD OF FROBENIUS

read 'frobrec.mpl':

#READ PROCEDURE FOR THE POWER SERIES METHOD ABOUT AN ORDINARY POINT

read 'pser.mpl':

# FINDS THE PARTICULAR INTEGRAL

read 'pisern.mpl':

# SET PRECISION

Digits:= 45;
Order := 50;

# DIFFERENTIAL EQUATION  FROM LATTA AND HESS (1973)
# RHS IS SET TO ZERO

de_h:=(s^2)*diff(G(s),s$2)+3*s*diff(G(s),s)-(s^2+s*coth(s))*G(s)=0:

# INCLUDES RHS

de_p:= (s^2)*diff(G(s),s$2)+3*s*diff(G(s),s)-(s^2+s*coth(s))*G(s)
       =-2*s*exp(-s):

#*************************************************************
```

```
# CALCULATES  SERIES EXPANSIONS FOR HOMOGENEOUS  SOLUTIONS
# ABOUT THE SINGULAR POINT  AT ORIGIN Gh_0
# AND ABOUT  THE ORDINARY POINTS s= 5/2, 4, 6 , 8
# Gh_52, Gh_4,  Gh_6, Gh_8

y:= Frobenius_Solution_2(de_h,G(s)):

Gh_0 := subs(_C1=1,_C2=0,y):
save Gh_0,'Gh_0.m';
#************************************************************************

Gh_52 := Power_Series([de_h,G(5/2)=evalf(subs(s=5/2,Gh_0)),
D(G)(5/2)=evalf(subs(s=5/2, diff(Gh_0,s)))],G(s),5/2):

save Gh_52,'Gh_52.m';
#************************************************************************

Gh_4 := Power_Series([de_h,G(4)=evalf(subs(s=4,Gh_52)),
        D(G)(4)=evalf(subs(s=4,diff(Gh_52,s)))],G(s),4):

save Gh_4,'Gh_4.m';
#************************************************************************

Gh_6:= Power_Series([de_h,G(6)=evalf(subs(s=6,Gh_4)),
D(G)(6)=evalf(subs(s=6,diff(Gh_4,s)))],G(s),6):

save Gh_6,'Gh_6.m';
#************************************************************************

Gh_8:= Power_Series([de_h,G(8)=evalf(subs(s=8,Gh_6)),
D(G)(8)=evalf(subs(s=8,diff(Gh_6,s)))],G(s),8):

save Gh_8,'Gh_8.m';
#************************************************************************
# CALCULATES  SERIES EXPANSIONS FOR  PARTICULAR INTEGRAL
# ABOUT THE SINGULAR POINT AT ORIGIN, Gp_0
# AND ABOUT  THE ORDINARY POINTS s= 5/2 , 4, 6 ,8.
# Gp_52, Gp_4,  Gp_6, Gp_8

Gp_0:= Pi_Series_N(de_p,G(s)):
```

```
save Gp_0, 'Gp_0.m';
#*************************************************************************

Gp_52:= Power_Series([de_p,G(5/2)=evalf(subs(s=5/2,Gp_0)),
D(G)(5/2)=evalf(subs(s=5/2, diff(Gp_0,s)))],G(s),5/2):

save Gp_52,'Gp_52.m';
#*************************************************************************

Gp_4:= Power_Series([de_p,G(4)=evalf(subs(s=4,Gp_52)),
D(G)(4)=evalf(subs(s=4,diff(Gp_52,s)))],G(s),4):

save Gp_4,'Gp_4.m';
#*************************************************************************

Gp_6:= Power_Series([de_p,G(6)=evalf(subs(s=6,Gp_4)),
D(G)(6)=evalf(subs(s=6,diff(Gp_4,s)))],G(s),6):

save Gp_6,'Gp_6.m';
#*************************************************************************

Gp_8:= Power_Series([de_p,G(8)=evalf(subs(s=8,Gp_6)),
D(G)(8)=evalf(subs(s=8,diff(Gp_6,s)))],G(s),8):

save Gp_8,'Gp_8.m';
#*************************************************************************

#   ASYMPTOTIC SOLUTION FOR s --> INFINITY

# FIND THE  HOMOGENIUS SOLUTION gh2 AND PARTICULAR SOLUTION gp2
# FOR LARGE s (2ND ORDER APPROXIMATION)

 coth_s:= 1+ 2*exp(-2*s):

# WE CALCULATE THE EXPANSION BY SUBSTITUTION THE COTH TERM.
# FIND THE HOMOGENIUS SOLUTION  gh2  FOR LARGE s

 subs( coth(s)=coth_s,de_h):
 subs( G(s)= exp(-s)/s^2+ f(s)*exp(-3*s),''):
```

```
 simplify(");
 simplify(expand( exp(3*s)*"));

# THE D.E. CONTAINS A TERM FROM THE NEXT ORDER THAT WE REMOVE

 subs( exp(-2*s) = 0,exp(2*s) = 0,"):
 dsolve(",f(s)):

#  WE DO NOT NEED THE GROWING SOLUTIONS.

 f(s):= subs({_C1=0,_C2=0},rhs(")):
#=================================================================

 gh:= exp(-s)/s^2;
 gh2:= gh + map(simplify,expand(exp(-3*s)*f(s)));
 save gh2,'gh2.m';
#=================================================================
#  FIND THE PARTICULAR INTEGRAL  gp2  FOR LARGE s (2ND ORDER)

 subs( coth(s)=coth_s,de_p):
 subs( G(s)=exp(-s)/2+ fp(s)*exp(-3*s),"):
 simplify(");
 simplify(expand( exp(3*s)*")):

# THE D.E. CONTAINS A TERM FROM THE NEXT ORDER THAT WE REMOVE
 subs( exp(-2*s) = 0,exp(2*s) = 0,"):
 dsolve(",fp(s)):

# AGAIN WE DO NOT NEED THE GROWING SOLUTIONS.
 fp(s):=subs( {_C1=0,_C2=0},rhs(")):
#=================================================================

 gp:= (1/2)*exp(-s);
 gp2:= gp + map(simplify,expand(exp(-3*s)*fp(s)));
 save gp2,'gp2.m';
```

```
#*********************************
#  2. FIND THE CONSTANTS K AND k
#*********************************

# SET PRECISION

Digits:=45;
Order:=50;

 read 'Gh_0.m':
 read 'Gh_52.m':
 read 'Gh_4.m':
 read 'Gh_6.m':
 read 'Gh_8.m':

 read 'Gp_0.m':
 read 'Gp_52.m':
 read 'Gp_4.m':
 read 'Gp_6.m':
 read 'Gp_8.m':

 read 'gh2.m';
 read 'gp2.m';

writeto(rlhkk):
 small_s := Gp_6 + k* Gh_6:
 large_s := gp^ + K *gh2:

 for i from 5.8 by .2 to 10 do
      q[i] := subs(s=i,small_s):
      Dq[i] := subs(s=i,diff(small_s,s)):
      Q[i] := subs(s=i,large_s):
      DQ[i] := subs(s=i,diff(large_s,s)):
      S1:=solve({q[i]=Q[i],Dq[i]=DQ[i]},{k,K}):
      assign(S1):
      print(i,k,K);
      k:='k':
      K:='K':
 od:
```

APPENDIX B
Computer Programs for Chapter 4

```
#-------------------------------------------------------------
# PROGRAM IS DIVIDED IN 4 PARTS:
#  1. FIND  SERIES SOLUTIONS ABOUT EXPANSIONS POINTS AND
#       ASYMPTOTIC SOLUTION FOR s --> INFINITY
#  2. FIND THE CONSTANTS C AND c
#  3. DEFINITE INTEGRAL IN g_s  (K1)
#  4. DEFINITE INTEGRAL IN g_w  (K2)
#-------------------------------------------------------------


#  1. FIND  SERIES SOLUTIONS ABOUT EXPANSIONS POINTS

# READ PROCEDURE FOR THE METHOD OF FROBENIUS

read 'frobrec.mpl':

#READ PROCEDURE FOR THE POWER SERIES METHOD ABOUT AN ORDINARY POINT

read 'pser.mpl':

# FINDS THE PARTICULAR INTEGRAL

read 'pisern.mpl':

interface(echo=2):

# SET PRECISION

Digits:=45;
Order:=75;

# DIFFERENTIAL EQUATION (78) FROM O'NEILL AND STEWARTSON (1967)

K  :=(1/s)-coth(s):
DK :=diff(K,s)    :
DDK:=diff(K,s$2)  :
DDX :=diff(coth(s)-1,s$2):
```

```
# RHS IS SET TO ZERO

de_h:=(s^3)*DK*diff(A(s),s$2)+s*diff(A(s),s)*((s^2)*DDK+3*s*DK+2*K)
        - A(s)*((s^2)*DDK+4*s*DK+2*K)=0:

# INCLUDES RHS

de_p:=(s^3)*DK*diff(A(s),s$2)+s*diff(A(s),s)*((s^2)*DDK+3*s*DK+2*K)
     - A(s)*((s^2)*DDK+4*s*DK+2*K)=-(s^2)*DDX:

#*********************************************************************
# CALCULATES  SERIES EXPANSIONS FOR HOMOGENEOUS  SOLUTIONS
# ABOUT THE SINGULAR POINT  AT ORIGIN Ah_0
# AND ABOUT  THE ORDINARY POINTS s= 5/2, 4, 6 , 8 AND 10
# Ah_52, Ah_4, Ah_6, Ah_8 AND Ah_10

y:= Frobenius_Solution_2(de_h,A(s)):

Ah_0 := subs(_C1=1,_C2=0,y):
save Ah_0,'Ah_0.m';
#*********************************************************************

Ah_52 := Power_Series([de_h,A(5/2)=evalf(subs(s=5/2,Ah_0)),
D(A)(5/2)=evalf(subs(s=5/2, diff(Ah_0,s)))],A(s),5/2):

save Ah_52,'Ah_52.m';
#*********************************************************************

Ah_4 := Power_Series([de_h,A(4)=evalf(subs(s=4,Ah_52))
        D(A)(4)=evalf(subs(s=4,diff(Ah_52,s)))],A(s),4):

save Ah_4,'Ah_4.m';
#*********************************************************************

Ah_6:= Power_Series([de_h,A(6)=evalf(subs(s=6,Ah_4)),
D(A)(6)=evalf(subs(s=6,diff(Ah_4,s)))],A(s),6):

save Ah_6,'Ah_6.m';
#*********************************************************************
```

```
Ah_8:= Power_Series([de_h,A(8)=evalf(subs(s=8,Ah_6)),
D(A)(8)=evalf(subs(s=8,diff(Ah_6,s)))],A(s),8):

save Ah_8,'Ah_8.m';
#*************************************************************************

Ah_10:= Power_Series([de_h,A(10)=evalf(subs(s=10,Ah_8)),
D(A)(10)=evalf(subs(s=10,diff(Ah_8,s)))],A(s),10):

save Ah_10,'Ah_10.m';
#*************************************************************************

# CALCULATES  SERIES EXPANSIONS FOR  PARTICULAR INTEGRAL
# ABOUT THE SINGULAR POINT AT ORIGIN, Ap_0
# AND ABOUT  THE ORDINARY POINTS s= 5/2 , 4, 6 ,8 AND 10 .
# Ap_52, Ap_4,  Ap_6, Ap_8  AND Ap_10

Ap_0:= Pi_Series_N(de_p,A(s)):

save Ap_0, 'Ap_0.m';
#*************************************************************************

Ap_52:= Power_Series([de_p,A(5/2)=evalf(subs(s=5/2,Ap_0)),
D(A)(5/2)=evalf(subs(s=5/2, diff(Ap_0,s)))],A(s),5/2):

save Ap_52,'Ap_52.m';
#*************************************************************************

Ap_4:= Power_Series([de_p,A(4)=evalf(subs(s=4,Ap_52)),
D(A)(4)=evalf(subs(s=4,diff(Ap_52,s)))],A(s),4):

save Ap_4,'Ap_4.m';
#*************************************************************************

Ap_6:= Power_Series([de_p,A(6)=evalf(subs(s=6,Ap_4)),
D(A)(6)=evalf(subs(s=6,diff(Ap_4,s)))],A(s),6):

save Ap_6,'Ap_6.m';
#*************************************************************************
```

```
Ap_8:= Power_Series([de_p,A(8)=evalf(subs(s=8,Ap_6)),
D(A)(8)=evalf(subs(s=8,diff(Ap_6,s)))],A(s),8):

save Ap_8,'Ap_8.m';
#*************************************************************************

Ap_10:= Power_Series([de_p,A(10)=evalf(subs(s=10,Ap_8)),
D(A)(10)=evalf(subs(s=10,diff(Ap_8,s)))],A(s),10):


save Ap_10,'Ap_10.m';
#*************************************************************************

# TO FIND THE DEFINITE INTEGRAL K1 AND K2 WE NEED THE SERIES OF THE
# FUNCTION WHICH CONTAINS NON-RATIONAL POWERS OF s.  MAPLE'S SERIES
# DATA STRUCTURE DOES NOT ACCEPT THAT.  HENCE IN THE ROUTINE
#'frobrec.mpl', (-2+sqrt(10)) IS REPLACED BY ITS RATIONAL VALUE
# WHICH IS APPROXIMATELY  14510839/12484830 AND THIS NEW ROUTINE
# IS NAMED  'f.mpl'.

read 'f.mpl':

FF:= Frobenius_Solution_2(de_h,A(s)):

Ah_0r := subs(_C1=1,_C2=0,FF):
save Ah_0r,'Ah_0r.m';
#*************************************************************************

#   ASYMPTOTIC SOLUTION FOR s --> INFINITY

# FIND THE  HOMOGENIUS SOLUTION ah2 AND PARTICULAR SOLUTION ap2
# FOR LARGE s (2ND ORDER APPROXIMATION)

 coth_s:= 1+ 2*exp(-2*s):

# WE CALCULATE THE EXPANSION BY SUBSTITUTION THE COTH TERM.
# FIND THE HOMOGENEOUS SOLUTION  ah2  FOR LARGE s

 subs( coth(s)=coth_s,de_h):
```

```
 subs( A(s)= exp(-2*s)+ f(s)*exp(-4*s),"):
 simplify("):
 simplify(expand( exp(4*s)*")):
 collect(",[diff(f(s),s$2),diff(f(s),s),f(s)]):
 subs(exp(-2*s)=0,exp(-4*s)=0,exp(-6*s)=0,"):
 dsolve(",f(s)):
#  WE DO NOT NEED THE GROWING SOLUTIONS.
 f(s):=subs( {_C1=0,_C2=0},rhs(")):
#=================================================================
 ah:=exp(-2*s);
 ah2:= ah+map(simplify,expand(exp(-4*s)*f(s)));
 save ah2,'ah2.m';

#=================================================================

#  FIND THE PARTICULAR INTEGRAL  ap2  FOR LARGE s (2ND ORDER)

 subs( coth(s)=coth_s,de_p):
 subs( A(s)= -2*(s^2)*exp(-2*s)+ fp(s)*exp(-4*s),"):
 simplify("):
 simplify(expand( exp(4*s)*")):
 collect(",[diff(fp(s),s$2),diff(fp(s),s),fp(s)]):
 subs(exp(-2*s)=0,exp(-4*s)=0,exp(-6*s)=0,"):
 dsolve(",fp(s)):
# AGAIN WE DO NOT NEED THE GROWING SOLUTIONS.
 fp(s):=subs( {_C1=0,_C2=0},rhs(")):
#=================================================================
 ap:= -2*(s^2)*exp(-2*s);
 ap2:=-2*(s^2)*exp(-2*s)+exp(-4*s)*fp(s);
 save ap2,'ap2.m';
#=================================================================
```

```
#**********************************
#  2. FIND THE CONSTANTS C AND c
#**********************************
# SET PRECISION
Digits:=45;
Order:= 75;

 read 'Ah_0.m':
 read 'Ah_52.m':
 read 'Ah_4.m':
 read 'Ah_6.m':
 read 'Ah_8.m':
 read 'Ah_10.m':

 read 'Ap_0.m':
 read 'Ap_52.m':
 read 'Ap_4.m':
 read 'Ap_6.m':
 read 'Ap_8.m':
 read 'Ap_10.m':

 read 'ap2.m':
 read 'ah2.m':

 small_s := Ap_8 + c* Ah_8:
 large_s := ap2 + C *ah2:

 for i from 8 by .2 to 12 do
      q[i] := subs(s=i,small_s):
      Dq[i] := subs(s=i,diff(small_s,s)):
      Q[i] := subs(s=i,large_s):
      DQ[i] := subs(s=i,diff(large_s,s)):
      S1:=solve({q[i]=Q[i],Dq[i]=DQ[i]},{c,C}):
      assign(S1):
      print(i,c,C);
      c:='c':
      C:='C':
 od:
```

```
#**********************************
#  3. DEFINITE INTEGRAL IN g_s (K1)
#**********************************

# SET PRECISION

Digits:=45;
Order:=75;

# FOR THE SOLUTION ABOUT s=0, WE READ 'Ah_0r.m' BECAUSE IT IS
# IN RATIONAL POWERS.

 read 'Ah_0r.m':
 read 'Ah_52.m':
 read 'Ah_4.m':
 read 'Ah_6.m':
 read 'Ah_8.m':
 read 'Ah_10.m':

 read 'Ap_0.m':
 read 'Ap_52.m':
 read 'Ap_4.m':
 read 'Ap_6.m':
 read 'Ap_8.m':
 read 'Ap_10.m':

# THESE VALUES OF c AND C ARE FOUND EARLIER

 c := 0.21836945458743559:
 C := 3.8797393774:

 ap:= -2*(s^2)*exp(-2*s):
 ah:= exp(-2*s):

# INTEGRAND FOR K1

 I_I := (A+(3/(5*s^2)))* ( 2*s*csch(s)^2 - (coth(s)-1)*(1+2*s+s^2*
        csch(s)^2) ) - ( 3*exp(-2*s)/5)* (1/s^3 + 1/s^2 + 2/(3*s)
 + (5*(2*s-1)/3)*coth(s) ) - (3/5)* ( 2/s^3 - (3/s^2 +2/s)*
 (coth(s)-1) ):
```

```
A_s1  := Ap_0 + c* Ah_0r:
A_s52 := Ap_52 + c* Ah_52:
A_s4  := Ap_4 + c* Ah_4:
A_s6  := Ap_6 + c* Ah_6:
A_s8  := Ap_8 + c* Ah_8:
A_s10 := Ap_10 + c* Ah_10:

A_L := ap + C *ah:

I_s1:=  subs(A=value(A_s1) ,I_I):
series(I_s1,s,Order):
subs(0=0,"):
simplify("):
S1:=convert(",polynom):

I_s52 :=  subs(A=value(A_s52) ,I_I):
I_s4  := subs(A=value(A_s4) ,I_I):
I_s6  :=  subs(A=value(A_s6) ,I_I):
I_s8  :=  subs(A=value(A_s8) ,I_I):
I_s10 :=  subs(A=value(A_s10) ,I_I):

# GIVES POLE AT INFINITY, HENCE coth(s) AND cosech(s) ARE TAKEN AS

subs({A=value(A_L),coth(s)=1+2*exp(-2*s),csch(s)=
                        2*exp(-s)+2*exp(-3*s)},I_I):
I_L  := expand(simplify(")):

K1_1  := (4/5)+(1/2)*evalf(Int(S1,s=0..2));
K1_52 := (1/2)*evalf(Int(I_s52,s=2..4));
K1_4  := (1/2)*evalf(Int(I_s4, s=4..6));
K1_6  := (1/2)*evalf(Int(I_s6, s=6..8));
K1_8  := (1/2)*evalf(Int(I_s8, s=8..10));
K1_10 := (1/2)*evalf(Int(I_s10, s=10..12));
K1_L  := (1/2)*evalf(Int(I_L,s=12..infinity));
K1    := K1_L + K1_1 + K1_52 + K1_4 + K1_6 + K1_8 + K1_10 ;
```

```
#************************************
#  4. DEFINITE INTEGRAL IN g_w  (K2)
#************************************

# SET PRECISION

Digits:=45;
Order:=75;

# FOR THE SOLUTION ABOUT s=0, WE READ 'Ah_0r.m' BECAUSE IT IS
# IN RATIONAL POWERS.

 read 'Ah_0r.m':
 read 'Ah_52.m':
 read 'Ah_4.m':
 read 'Ah_6.m':
 read 'Ah_8.m':
 read 'Ah_10.m':

 read 'Ap_0.m':
 read 'Ap_52.m':
 read 'Ap_4.m':
 read 'Ap_6.m':
 read 'Ap_8.m':
 read 'Ap_10.m':

 writeto(rnsK2):
interface(echo=2):

# THESE VALUES OF c AND C ARE FOUND EARLIER


 c := 0.21836945458743559:
 C := 3.8797393774:

 ap:= -2*(s^2)*exp(-2*s):
 ah:= exp(-2*s):
 K := (1/s)-coth(s):
```

```
# INTEGRAND FOR K2

I_I := 4*s*A+ (s^2)*DDA*K+2*(coth(s)-1)-((4*exp(-2*s))/(5*s)):

small_s1  := Ap_0 + c* Ah_0r:
small_s52 := Ap_52 + c* Ah_52:
small_s4  := Ap_4 + c* Ah_4:
small_s6  := Ap_6 + c* Ah_6:
small_s8  := Ap_8 + c* Ah_8:
small_s10 := Ap_10 + c* Ah_10:

A_s1   := small_s1:
DA_s1  := diff(A_s1,s):
DDA_s1 := diff(A_s1,s$2):

A_s52  := small_s52:
DA_s52 := diff(A_s52,s):
DDA_s52:= diff(A_s52,s$2):

A_s4   := small_s4:
DA_s4  := diff(A_s4,s):
DDA_s4 := diff(A_s4,s$2):

A_s6   := small_s6:
DA_s6  := di f(A_s6,s):
DDA_s6 := diff(A_s6,s$2):

A_s8   := small_s8:
DA_s8  := diff(A_s8,s):
DDA_s8 := diff(A_s8,s$2):

A_s10   := small_s10:
DA_s10  := diff(A_s10,s):
DDA_s10 := diff(A_s10,s$2):


large_s := ap + C *ah:

A_L    := large_s:
DA_L   := diff(A_L,s):
```

```
DDA_L  := diff(A_L,s$2):

#  GIVES POLE AT ZERO, HENCE SERIES EXPANSION ABOUT s=0 IS USED.

subs({A=value(A_s1),DDA=value(DDA_s1)},I_I):
series(",s,Order):
subs(O=0,"):
simplify("):

I_s1:=convert(",polynom):
I_s52 := subs({A=value(A_s52),DDA=value(DDA_s52)},I_I):
I_s4  := subs({A=value(A_s4),DDA=value(DDA_s4)},I_I):
I_s6  := subs({A=value(A_s6),DDA=value(DDA_s6)},I_I):
I_s8  := subs({A=value(A_s8),DDA=value(DDA_s8)},I_I):
I_s10 := subs({A=value(A_s10),DDA=value(DDA_s10)},I_I):
I_s12 := subs({A=value(A_s12),DDA=value(DDA_s12)},I_I):

subs({A=value(A_L),DDA=value(DDA_L),coth(s)=1+2*exp(-2*s)} ,I_I):
I_L  := expand(simplify(")):

K2_1 := (1/5) + (1/4)*evalf(Int(I_s1,s=0..2));

K2_52:= (1/4)*evalf(Int(I_s52,s=2..4));

K2_4 := (1/4)*evalf(Int(I_s4,s=4..6)) ;

K2_6 := (1/4)*evalf(Int(I_s6,s=6..8)) ;

K2_8 := (1/4)*evalf(Int(I_s8,s=8..10)) ;

K2_10:= (1/4)*evalf(Int(I_s10,s=10..12)) ;

K2_L:=(1/4)*evalf(Int(I_L,s=12..infinity));

K2 := K2_L + K2_1 + K2_52 + K2_4 + K2_6+ K2_8+ K2_10;
```

APPENDIX C

The ratio of the coefficients in the series solution.

$2n$	$\lvert a_{2n+2}/a_{2n} \rvert$
50	.08700671127973
52	.08746806805313
54	.08791643451581
56	.08834558476605
58	.08874909745947
60	.08912666772116
62	.08948118793328
64	.08981558921275
66	.09013189009876
68	.09043148687546
70	.09071555876232
72	.09098522719009

APPENDIX D
Computer Program for Chapter 5

```
#*******************************#
#          Solve95              #
#*******************************#

interface(verboseproc=0):
#*******************************************************************
# MAIN PROCEDURE
#*******************************************************************

# GIVEN A LIST OF ALL THE ROOTS, PROCEDURE All_Roots SORTS THE
# INDEPENDENT ROOTS AND GIVES THEIR NUMERICAL VALUES
# (THROUGH allvalues)

Solve95 := proc(S)
            local S2, i, p, Set1, ldeg, hdeg, cdeg, ll, ii,
              m, n, k, flag, nset, Bindep, Bdep, j, allvalues_set,
               nset, rootofs,indep, Dep, num_set, r, deg, p_x,
               aterm, mterm, Sa, Sm, rho, mrest, Sm_set;

# S    : LIST OF THE ROOTS (INPUT)

# SEPARATE THE ROOTS WHICH HAVE Rootof IN THEM.
# rootofs.0 IS THE LIST OF THE ROOTS NOT CONTAINING RootOf.
# S2 IS THE LIST OF ALL THE ROOTS CONTAINING Rootof.

 rootofs.0 := []:
 S2 := []:
 for i in S do
     if has(i,RootOf) then
          S2 := [op(S2),i];
     else
          rootofs.0 := [op(rootofs.0),i]:
     fi;
 od:

# SORT THE ROOTS ACCORDING TO THEIR DEGREE (IN ASCENDING ORDER)
# ------------------------------------------------------------
```

```
# START WITH THE FIRST MEMBER OF THE  LIST

 p := op(1,S2):
 Set1 := [p]:

# CURRENTLY, LOWER AND HIGHER DEGREE IN THE LISTS ARE SAME AS THE
# DEGREE OF THE FIRST (ONLY) MEMBER.

 ldeg := degree(op(p)):
 hdeg := degree(op(p)):

# COMPARE THE DEGREE OF OTHER ROOTS, cdeg, TO THE LOW (ldeg) AND
# THE HIGH (hdeg).

 for i from 2 to nops(S2) do
      cdeg := degree(op(op(i,S2))):

# IF THE DEGREE OF THE CURRENT ROOT IS LESS THAN THE LOWER DEGREE
# OF THE ROOTS IN THE LIST, PUT THE CURRENT ROOT AT THE BEGINNING.
# NOW THIS WILL BE THE LOWER DEGREE OF THE UPDATED LIST.

      if cdeg < ldeg then
           Set1 := [op(i,S2),op(Set1)];
           ldeg := cdeg;

# IF THE DEGREE OF THE CURRENT ROOT IS MORE THAN THE LOWER DEGREE
# OF THE ROOTS IN THE LIST, PUT THE CURRENT ROOT AT THE END.
# NOW THIS WILL BE THE HIGHER DEGREE OF THE UPDATED LIST.

      elif cdeg > hdeg then
           Set1 := [op(Set1),op(i,S2)];
           hdeg := cdeg;

# IF THE DEGREE OF THE CURRENT ROOT IS IN BETWEEN OF THE DEGREES
# OF THE ROOTS IN THE LIST Set1, FIND OUT THE ROOT IN THE LIST
# FOR WHICH THE CURRENT DEGREE IS MORE THAN ITS DEGREE.

      else
           ll := 0:
```

```
            for ii from 1 to nops(Set1) do
                if cdeg > degree(op(op(ii,Set1))) then ll:= ll+1 fi;
            od;

# THEN PUT THE CURRENT ROOT AFTER THIS ll'th ROOT.

            Set1:= [seq(op(m,Set1),m=1..ll),op(i,S2),seq(op(n,Set1),
                        n=ll+1..nops(Set1))];
        fi;
 od:
 m:= 'm':

# SORTING OF THE ROOTS IN ASCENDING  ORDER IS COMPLETE ---^
# ------------------------------------------------------

# PUT ALL THE ROOTS WITH LIKE DEGREES TOGETHER
# IN SEPARATE LISTS rootof.m

 for m from 1 to nops(Set1) do
     rootofs.m := []:
 od:

# COMPARE THE DEGREE OF SORTED ROOTS BETWEEN CONSECUTIVE MEMBERS

 m := 1:
 for k from 1 to nops(Set1)-1 do

# IF THE DEGREE OF TWO ROOTS IS SAME, PUT THEM TOGETHER

     if degree(op(op(k,Set1)))=degree(op(op(k+1,Set1))) then
         rootofs.m := [op(rootofs.m),op(k,Set1)];

# OTHERWISE PUT IT IN A SEPARATE LIST.  SINCE THE ROOTS ARE ALREADY
# SORTED IN ASCENDING ORDER, A ROOT WITH A LOWER DEGREE THAN THE
# CURRENT ONE WILL NOT APPEAR LATER, SO INCREASE THE COUNTER m.

     else
         rootofs.m := [op(rootofs.m),op(k,Set1)];
         m := m+1:
     fi;
```

```
 od:

# AN EMPTY LIST rootof.m (FOR THE CURRENT m) INDICATES THAT
# LISTS UPTO rootofs.(m-1) HAVE ROOTS IN THEM.

 if rootofs.m = [] then m := m-1 ; fi;

# WE STILL HAVE THE LAST ROOT OF THE LIST Set1.  COMPARE IT'S DEGREE
# WITH ALL THE OTHER LISTS AND DETERMINE WHERE IT SHOULD GO.
# THE VALUE OF flag WILL INDICATE IF THE ROOT WAS ABLE TO FIND
# A LIST WITH LIKE DEGREE.

 flag := 0:
 for n from 1 to m do
     if degree(op(op(nops(Set1),Set1)))=degree(op(op(1,rootofs.n)))
     then
         rootofs.n := [op(rootofs.n),op(nops(Set1),Set1)];
         flag := 1:
     fi;
 od;

# IF THE LAST ROOT DID NOT FIND ANY LIST WITH LIKE DEGREE,
# THEN PUT IT IN A SEPARATE LIST.

 if flag = 0 then rootofs.(m+1) := [op(nops(Set1),Set1)]; fi:
 n := 'n':  m := 'm':

# DETERMINE HOW MANY LISTS ARE MADE WITH THE ROOTS.
# THIS NUMBER IS nset.

 nset := 0:
 for n from 1 to nops(Set1) do
     if not rootofs.n = [] then
         nset := nset+1:
     fi;
 od:

#*************************************************
 lprint('COMPLETE SET OF ROOTS OF THE GIVEN POLYNOMIAL IS');
```

```
print(rootofs.0);  i := 'i':

for i from 1 to nset do

# Sm_set IS A SET OF ALL INDEPENDENT CANONICAL FORMS ON WHICH
# 'fsolve' IS  TO BE APPLIED.

   Sm_set := {}:

   for j from 1 to nops(rootofs.i) do

# p_x ARE IRREDUCIBLE POLYNOMIALS TO BE REDUCED TO CANONICAL FORM
# USING THEOREM 4 AND 6, IN PARTICULAR THESE ARE THE ARGUMENT OF
# RootOf'S.

      p_x := op(op(j,rootofs.i));
      deg := degree(p_x);

# REDUCE THE POLYNOMIAL SO THAT THE LEADING COEFFICIENT IS 1.

      p_x := simplify(p_x/coeff(p_x,_Z,deg));

# FIND THE ADDITIVE TERM AND THE CORRESPONDING REDUCED POLYNOMIAL
# USING THEOREM 4.

      aterm := - simplify((coeff(p_x,_Z,deg-1)/deg));
      Sa:= simplify(subs(_Z = _Z + aterm, p_x));

# FIND THE MULTIPLICATIVE TERM AND THE CORRESPONDING
# REDUCED POLYNOMIAL USING THEOREM 6.

      mterm := simplify(coeff(Sa,_Z,0)^(1/deg)):

      if type(mterm,rational) then
         Sm.j := simplify(subs(_Z = _Z * mterm, Sa)/mterm^deg);

# IF mterm IS NOT RATIONAL, THEN USE mterm = rho * b_0^(1/N)
# WHERE b_0 IS THE CONSTANT TERM OF THE POLYNOMIAL IN CANONICAL
# FORM.
```

```
      elif type(mterm,'*') then
         rho := op(1,mterm):
         mrest := prod('op(i,mterm)','i' = 2..nops(mterm)):
         if type(rho,rational) then
            Sm.j := simplify(subs(_Z = _Z * rho, Sa)/rho^deg):
            mterm := rho:
         else
            Sm.j := simplify(subs(_Z = _Z * mterm, Sa)/mterm^deg):
         fi;
      elif type(mterm,'^') then
         mterm := 1:
         Sm.j := simplify(subs(_Z = _Z * mterm, Sa)/mterm^deg):
      else
         Sm.j := simplify(subs(_Z = _Z * mterm, Sa)/mterm^deg):
      fi;
      aterm + mterm * RootOf(Sm.j);
      print(");

# UPDATE THE SET OF INDEPENDENT CANONICAL FORMS

      Sm_set := {op(Sm_set)} union {Sm.j}:
   od:

# FIND THE NUMERICAL VALUE OF INDEPENDENT ROOTS ONLY
lprint('where');
   for k from 1 to nops(Sm_set) do
      fsolve(op(k,Sm_set),_Z,complex);
      map(proc(r,a,b) RootOf(a,b,r) end,{"},op(k,Sm_set),_Z);
      print(");
   od:
 od:
end:
#*************************************************************************
```

APPENDIX E
Computer Programs for Chapter 6

```
#*******************************#
#        CLASSIFY POINT         #
#*******************************#

# THIS PROCEDURE FINDS THE REGULAR AND IRREGULAR SINGULAR POINTS
# OF A NTH ORDER LINEAR ORDINARY DIFFERENTIAL EQUATION.

interface(verboseproc=0):
Classify_Point := proc(deqn,var)
                  local ord,R,indep_var,P,k,S,Singular_points,a1,
                      r, b, S_regular, S_irregular, Sol, j, Cond1:

# INPUT
# deqn : pN(x)*diff(y(x),x$N)+...+p1(x)*diff(y(x),x)+p0(x)*y(x) = 0
# var  : DEPENDENT VARIABLE OF INDEPENDENT VARIABLE, e.g. y(x)

 indep_var := op(var):  # EXTRACT INDEPENDENT VARIABLE

# CALCULATE ORDER OF THE ODE BY INCREMENTING ORDER UNTIL IT
# REACHES THE HIGHEST DIFFERENTIATION IN THE ODE.

 for ord from 0 while has(lhs(deqn),diff(var,indep_var$ord+1 )) do
 od:

# EXTRACT COEFFICIENTS P(k)

 R := subs( seq(diff(var,indep_var$(ord-k+1))=D[ord-k+1],k=1..ord)
                                              ,lhs(deqn)):

 k:= 'k':
 for k from 1 to ord do
    P[k] := coeff(R,D[k]);
 od:
 P[0] := coeff(R,var);

 k:= 'k':
 for k from 0 to ord do
```

```
    S[k] := convert(series(P[k],indep_var,Order+ord),polynom);
    b[k] := ldegree(S[k],indep_var):
 od:

 r:=min(seq(b[k],k=0..ord));

# FIND SINGULAR POINTS OF THE ODE

 k:= 'k':
 if type(P[ord]/indep_var^r,constant) then
     RETURN('No singular point');
 else
     a1 := P[ord]/indep_var^r;
     Singular_points := {solve(a1,indep_var)};
     if has(Singular_points,RootOf) or Singular_points={} then
        Digits := Digits+10;
        Singular_points:= {fsolve(a1,indep_var,complex,fulldigits)};
        Digits := Digits-10;
     fi;
 fi:

# IF Singular_points IS AN EMPTY SET

 if nops(Singular_points)=0 then
        RETURN('No singular point');
 fi:

 k := 'k':

 S_irregular := {}:

# CLASSIFY THE SINGULARITY AS REGULAR OR IRREGULAR
 for j in Singular_points do
     for k from 0 to ord-1 do
        Cond1:= limit(((indep_var-j)^(ord-k))*P[k]/P[ord]
                                              ,indep_var=j);
        if Cond1 = undefined or Cond1 = infinity then
            S_irregular:= S_irregular union {j};
            break;
        fi:
```

```
    od:
od:

k := 'k':

S_regular := Singular_points minus S_irregular:

if S_regular = {} then
   lprint('No regular singular point');
else
   lprint('Regular singular points');
   print(S_regular);
fi:

if S_irregular = {} then
   lprint('No irregular singular point');
else
   lprint('Irregular singular points');
   print(S_irregular);
fi:

# IN THE PROCEDURE FOR THE METHOD OF FROBENIUS, SERIES SOLUTION IS
# FOUND AROUND ORIGIN.

if S_regular <> {} and has(S_regular,0) = false  then
  lprint('None of the regular singular points is zero, therefore');
  lprint('rewrite the differential equation in terms of shifted
              coordinates');
  lprint('u= x-x0, x0 being the non-zero regular singular point');
  lprint('near which the solution is desired.');
fi:
end:
```

```
                    #*******************************#
                    #     POWER SERIES METHOD       #
                    #   ABOUT AN ORDINARY POINT     #
                    #(FOR A Nth ORDER LINEAR ODE)#
                    #*******************************#

 interface(verboseproc=0):

# PROCEDURE FOR COMPUTING Z TO THE N FALLING.

     Falling := proc(z,alpha)
                local k:
                if type(alpha,integer) then
                     product(z-k,k=0..(alpha-1))
                fi:
     end:

# PROCEDURE FOR THE POWER SERIES METHOD FOR SOLVING Nth ORDER LINEAR
# ORDINARY DIFFERENTIAL EQUATION ABOUT AN ORDINARY POINT.

Power_Series := proc(Deqn,var,pt)
              local deqn,ord,i,j,k,l,m,n,P,Q,y,dep_var,indep_var,
              ls_deqn,rs_deqn,pdummy,solution,Casefloat,first_sum,
              second_sum,inner_sum:

# THIS ROUTINE EXPECTS THE ODE IN THE FORM
# deqn : pN(x)*diff(y(x),x$N)+...+p1(x)*diff(y(x),x)+p0(x)*y(x) = 0
# var  : DEPENDENT VARIABLE OF INDEPENDENT VARIABLE, e.g. y(x)
# IF THE BC's ARE SUPPLIED THEY HAVE TO BE IN THE FORM OF LIST
# IN PROPER ORDER, e.g.
# Deqn: [deqn, y(pt)=.., D(y)(pt)=..,(D@@2)(y)(pt)=...]

 indep_var := op(var):  # EXTRACT INDEPENDENT VARIABLE, x
 dep_var := op(0,var):  # EXTRACT DEPENDENT VARIABLE, y

# EXTRACT DIFFERENTIAL EQUATION

 if type(Deqn,list) then deqn := op(1,Deqn) else deqn := Deqn fi:
 ls_deqn := lhs(deqn):  # LHS OF THE ODE
```

```
# CALCULATE ORDER OF THE ODE BY INCREMENTING ORDER UNTIL IT
# REACHES THE HIGHEST DIFFERENTIATION IN THE ODE.

 for ord from 0 while has(ls_deqn,diff(var,indep_var$ord+1 )) do
 od:

# TO CHECK IF ANY OF THE INITIAL CONDITIONS IS GIVEN IN
# FLOATING POINT

 Casefloat := false:
 if type(Deqn,list) then
     for l from 2 to ord+1 do
         if type( op(2,op(l,Deqn)), float) then
              Casefloat := true:
              break:
         fi:
     od:
 fi:

# EXTRACT COEFFICIENTS p_i
# IF B.C. ARE FLOATING POINT, APPLY 'evalf' TO THE 'pt' TO SPEED UP
# OTHERWISE LEAVE IT AS SUCH.

 Q := subs( seq(diff(var,indep_var$(ord-i+1))=D[ord-i+1],i=1..ord)
                                                        ,ls_deqn):

 i:= 'i':  l:= 'l':

 if Casefloat=true then
     for l from 1 to ord do
         pdummy:= convert(series(subs(indep_var=indep_var+evalf(pt),
                       coeff(Q,D[l])),indep_var,Order+ord),polynom):
         for m from 0 to Order-1 do
                     P[l,m] := coeff(pdummy,indep_var,m):
         od:
     od:

     pdummy := convert(series(subs(indep_var=indep_var+evalf(pt),
                     coeff(Q,var)),indep_var,Order+ord),polynom):
```

```
    for m from 0 to Order-1 do
            P[0,m] := coeff(pdummy,indep_var,m):
    od:

# R.H.S. OF THE DIFFERENTIAL EQUATION

    rs_deqn := convert(series(subs(indep_var=indep_var+evalf(pt)
                    ,rhs(deqn)),indep_var,Order+ord),polynom);

# IF rs_deqn DOES NOT HAVE A TAYLOR SERIES EXPANSION ABOUT THE GIVEN
# POINT THEN SET rs_deqn TO ZERO, SO THAT THE PROCEDURE RETURNS ONLY
# THE COMPLEMENTARY FUNCTIONS.  FOR THE PARTICULAR INTEGRAL, USER
# SHOULD CALL THE ROUTINE 'Variation_of_Parameters'.

    if type(rs_deqn,polynom)=false then
        rs_deqn := 0;
    fi;

 else
    for l from 1 to ord do
        pdummy := convert(series(subs(indep_var=indep_var+pt,
                    coeff(Q,D[l])),indep_var,Order+ord),polynom):
        for m from 0 to Order-1 do
            P[l,m] := coeff(pdummy,indep_var,m):
        od:
    od:

    pdummy := convert(series(subs(indep_var=indep_var+evalf(pt),
                    coeff(Q,var)),indep_var,Order+ord),polynom):

    for m from 0 to Order-1 do
        P[0,m] := coeff(pdummy,indep_var,m):
    od:

# R.H.S. OF THE DIFFERENTIAL EQUATION

    rs_deqn := convert(series(subs(indep_var=indep_var+pt
                ,rhs(deqn)),indep_var,Order+ord),polynom);

# IF rs_deqn DOES NOT HAVE A TAYLOR SERIES EXPANSION ABOUT THE GIVEN
```

```
# POINT THEN SET rs_deqn TO ZERO, SO THAT THE PROCEDURE RETURNS ONLY
# THE COMPLEMENTARY FUNCTIONS.  FOR THE PARTICULAR INTEGRAL, USER
# SHOULD CALL THE ROUTINE 'Variation_of_Parameters'.

     if type(rs_deqn,polynom)=false then
         rs_deqn := 0;
     fi;

 fi:

#*****************************************************
# FIND COEFFICIENTS y[k] THROUGH RECURRENCE RELATION

 i := 'i':  m := 'm':  l := 'l':
 for k from ord to Order do
     first_sum := 0:
     for j from (ord-1) to (k-1) do
         first_sum := first_sum + Falling(j,ord)
                        *P[ord,k-j]*y[j];
     od;
     second_sum := 0:
     for l from 0 to (ord-1) do
         inner_sum := 0:
         for i from 1 to (k-ord+1) do
             inner_sum := inner_sum + Falling(i,l) *y[i]
                                    *P[l,(k+1-ord-i)]
         od;
         second_sum := second_sum+inner_sum;
     od;

     y[k]:= (coeff(rs_deqn,indep_var,(k-ord))-first_sum
               -second_sum)/(P[ord,0]*Falling(k,ord) );
 od;

 l := 'l':  m := 'm':  n := 'n':  Q := 'Q':

# FIND POWER SERIES SOLUTION

 if type(Deqn,list) then
     solution:= subs({indep_var=indep_var-pt,y[0]=op(2,op(2,Deqn)),
```

```
                seq(y[l] =op(2,op(l+2,Deqn))/l!,l=1..ord-1)},
                          (sum(y[n]*(indep_var^n),n=0..Order-1))):
 else
     solution:= collect(subs({indep_var=indep_var-pt,y[0]=_C.1,
                seq(y[l] =_C.(l+1)/l!,l=1..ord-1)},(sum(y[n]*
                (indep_var^n),n=0..Order-1))),[seq(_C.(m)
                                                ,m=1..ord)]):
 fi:
 solution ;
end:
```

```
                    #******************************#
                    #    METHOD OF FROBENIUS       #
                    # FOR A NTH ORDER LINEAR ODE   #
                    #******************************#

interface(verboseproc=0):

# PROCEDURE FOR COMPUTING z falling alpha.

    Falling := proc(z,alpha)
               local k,prod1:
               if type(alpha,integer) then
                    prod1 := 1:
                    for k from 0 to alpha-1 do
                        prod1 := prod1 * (z-k):
                    od:
               fi:
                prod1;
               end:

# PROCEDURE TO COMPUTE product F(r+i) etc.

  Fri := proc(r,Q,ord,mn,mx)
         local prod1, sum1, i, m:
         prod1 := 1:
         for i from mn to mx do
             sum1 := 0:
             for m from 0 to ord do
                 sum1 := sum1 + Falling(r+i,m)*Q[ord-m,0]:
             od:
             prod1 := prod1 * sum1:
          od:
          expand(prod1);
   end:

# PROCEDURE FOR THE METHOD OF FROBENIUS FOR SOLVING
# Nth ORDER LINEAR ORDINARY DIFFERENTIAL EQUATION
# WITH REGULAR SINGULAR POINT AT ORIGIN.

Frobenius_Solution_N := proc(deqn,var)
```

```
            local indep_var,ord,Q,i,j,k,l,m,n,q,R,y,indic_eq,
            indic_roots,frob_sol,term1,sol_r,p_null,inner_sum,
            outer_sum,lower_sum,Qdummy,Map,Coffi,tmp1,left_sum,
             Y,m1,indic_sol:

# deqn : pN(x)*diff(y(x),x$N)+...+p1(x)*diff(y(x),x)+p0(x)*y(x) = 0
# var  : DEPENDENT VARIABLE OF INDEPENDENT VARIABLE, e.g. y(x)

 indep_var := op(var):  # EXTRACT INDEPENDENT VARIABLE, x

# CALCULATE ORDER OF THE ODE BY INCREMENTING ORDER UNTIL IT
# REACHES THE HIGHEST DIFFERENTIATION IN THE ODE.

 for ord from 0 while has(lhs(deqn),diff(var,indep_var$ord+1 )) do
 od:

# EXTRACT COEFFICIENTS Q(i) ( = x^i*p(ord-i)/p(ord) )

 R := subs( seq(diff(var,indep_var$(ord-i+1))=D[ord-i+1],i=1..ord)
                                                    ,lhs(deqn)):
 i:= 'i':
 Q[0,0] := 1:
 for i from 1 to ord-1 do
     Qdummy[i] := convert(series((coeff(R,D[ord-i])/coeff(R,D[ord]))
                       *(indep_var^i),indep_var,Order+ord),polynom):
 od:
 for m from 0 to Order do
     for i from 1 to ord-1 do
         Q[i,m] := coeff(Qdummy[i],indep_var,m):
     od:
 od:

 Qdummy[ord] := convert(series((coeff(R,var)/coeff(R,D[ord]))
                 *(indep_var^ord),indep_var,Order+ord),polynom):
 for m from 0 to Order do
     Q[ord,m] := coeff(Qdummy[ord],indep_var,m):
     if (m<>0) then Q[0,m] := 0 `i:
 od:

 i:= 'i':  m := 'm':
```

```
# INDICIAL EQUATION

 indic_eq:= sum('Falling(r,m)*Q[ord-m,0]','m'=0..ord):

# SOLUTION OF THE INDICIAL EQUATION

 indic_sol := solve(indic_eq,r);

# INDICIAL EQUATION IS AN IRREDUCABLE POLYNOMIAL,

# IF 'solve' DOES NOT FIND ALL THE ROOTS EXPLICITLY OR IS UNABLE TO
# FIND ANY SOLUTION, THEN PROCEDURE 'Solve95' IS TO BE USED.

 if has({indic_sol},RootOf) or {indic_sol}= {} then
      indic_sol := Solve95([solve(indic_eq,r)]):
 fi;

 if indic_sol = [] then RETURN(FAIL) fi;
 indic_roots := [];
 for i in indic_sol do
     for j to nops(indic_roots) do
         if type(i-op(1,op(j,indic_roots)),'integer') then
             indic_roots := subsop(j = [op(op(j,indic_roots)),i],
                                   indic_roots);
             break;
         fi;
     od;
        if nops(indic_roots) < j then
             indic_roots := [op(indic_roots),[i]];
        fi;
 od:

# ARRANGE ROOTS OF THE INDICIAL EQUATION

 indic_roots := sort(map(sort,indic_roots,proc(x,y) evalb(y < x)
                    end),proc(x,y) evalb(nops(x) < nops(y)) end);

 j := 'j':  l := 'l':  m := 'm':
```

```
#*****************************************************
# FIND COEFF'CIENTS y[k] THROUGH RECURRENCE RELATION

 Y[0] := 1:
 for k from 1 to Order do
     left_sum := 0:
     for m from 0 to ord do
         left_sum := left_sum+Falling(r,m)*Q[ord-m,k];
     od;
     outer_sum := 0:
     for l from 1 to k-1 do
        inner_sum := 0:
         for m1 from 0 to ord do
             inner_sum := inner_sum+Falling(r+l,m1)*Q[ord-m1,k-l];
         od:
         outer_sum := outer_sum+expand(Y[l]*Fri(r,Q,ord,l+1,k-1)
                                       *inner_sum);
     od:
     Y[k] := - simplify(Fri(r,Q,ord,1,k-1)*left_sum + outer_sum):
 od:
 i := 'i':  j := 'j':  k := 'k':  l := 'l':  m := 'm':  m1 := 'm1':
 inner_sum:='inner_sum': outer_sum:='outer_sum':
 lower_sum:='lower_sum':

 y[0]:=1:
 for k from 1 to Order do
     y[k] := Y[k] / Fri(r,Q,ord,1,k):
 od:
 i := 'i':   k := 'k':  m := 'm':

# FIND SOLUTION CORRESPONDING TO THE ROOTS OF THE INDICIAL EQUATION

 l := 1;
 frob_sol := 0;
 for i in indic_roots do
     term1 := product('subs(r = r+m,indic_eq)'
                 ,'m' = 1 .. i[1]-i[nops(i)]);
     inner_sum:= 0;
     for m from 0 to Order-1 do
         inner_sum:= inner_sum+ term1*y[m]*indep_var^m :
```

```
    od:
    sol_r := map(normal,convert(series(inner_sum
                  ,indep_var),polynom)):
    sol_r := indep_var^r*sol_r;
    n := NULL;
    for j in i do
        if j = n then next fi;
            n := j;
            p_null := nops(map(proc(x,j) if x = j then 1 else NULL
                                        fi end,i,j));
            if has(sol_r,ln(indep_var)) then
                  frob_sol := frob_sol + _C.(l)*collect(expand
                              (subs(r=j,sol_r)),[ln(indep_var)]):
            else
                  Map:= map(factor,subs(r=j,inner_sum)):
                  Coffi:= coeff(",indep_var,0):
                  tmp1:= map(proc(a,b) a/b end,Map,Coffi);
                  frob_sol := frob_sol + _C.(l)*indep_var^j*tmp1:
            fi:

            for k from 2 to p_null do
                sol_r := diff(sol_r,r);
                frob_sol := frob_sol+_C.(l+k-1)*collect(expand
                              (subs(r = j,sol_r)),[ln(indep_var)]):
            od;
            l := l+p_null;
            if nops(i) > 1 then
                  if l <= ord then
                  sol_r:=diff(sol_r,r):
            fi;
        fi;
    od:
od:
frob_sol;
end:
```

```
                    #*******************************#
                    #    METHOD OF FROBENIUS        #
                    # FOR A 2ND ORDER LINEAR ODE    #
                    #*******************************#
interface(verboseproc=0):
# PROCEDURE FOR THE METHOD OF FROBENIUS FOR SOLVING 2ND ORDER LINEAR
# ORDINARY DIFFERENTIAL EQUATION WITH REGULAR SINGULAR POINT
# AT ORIGIN.

Frobenius_Solution_2 := proc(deqn,var)
       local indep_var,R,Q1,Q2,i,m,indic_sol,r,Cond1,r1,r2,y,ind_eq,
              upper_term,sol1,sol2,sol,sol_set,sdummy,CC,n,j,K1,
              z,Qdummy1,Qdummy2,de1,K2,c,N,lower_term,upper_term2:

# deqn : p2(x)*diff(y(x),x$2)+p1(x)*diff(y(x),x)+p0(x)*y(x) = 0
# var  : DEPENDENT VARIABLE OF INDEPENDENT VARIABLE, e.g. y(x)

 indep_var := op(var):  # EXTRACT INDEPENDENT VARIABLE, x

# EXTRACT COEFFICIENTS Q(i) ( = x^i*p_(2-i)/p_2 )

 R := subs( seq(diff(var,indep_var$(3-i))=D[3-i],i=1..2),lhs(deqn)):
 i:= 'i':
 Qdummy1 := convert(series(indep_var*coeff(R,D[1])/coeff(R,D[2])
                    ,indep_var,Order+2),polynom);
 for m from 0 to Order do
     Q1[m] := coeff(Qdummy1,indep_var,m);
 od:

 Qdummy2 := convert(series((indep_var^2)*coeff(R,var)/coeff(R,D[2])
                    ,indep_var,Order+2),polynom);
 for m from 0 to Order do
     Q2[m] := coeff(Qdummy2,indep_var,m);
 od:

# PUT THE EQUATION IN STANDARD FORM FOR THE METHOD OF FROBENIUS.

 de1 := indep_var^2*diff(var,indep_var$2) + indep_var*Qdummy1*
        diff(var,indep_var) + Qdummy2*var:
 m := 'm':
```

```
# INDICIAL EQUATION

 ind_eq := r*(r-1) + Q1[0]*r + Q2[0];
 indic_sol := solve(",r);
 if has({indic_sol},RootOf) or {indic_sol}= {} then
      Digits := Digits+10;
      indic_sol := fsolve(ind_eq,r,complex,fulldigits);
      Digits := Digits-10;
 fi;
 if indic_sol = [] then RETURN(FAIL) fi;

# LABEL THE ROOTS AS r1 & r2 SUCH THAT [Re(r1) >= Re(r2)]

  Cond1 := is(Re(indic_sol[2]) > Re(indic_sol[1])):
  if Cond1=true then
    r1 := indic_sol[2]:    r2 := indic_sol[1]:
  else
    r1:=indic_sol[1]:      r2:=indic_sol[2]:
  fi:
  Cond1 := 'Cond1';

# SOLUTION CORRESPONDING TO r=r1.
#*********************************************************
# FIND COEFFICIENTS y[i] THROUGH RECURRENCE RELATION

 r:=r1;
 y[0]:=1:
 for i from 1 to Order do
     upper_term := 0:
     for m from 0 to i-1 do
          upper_term := upper_term + y[m]*((r+m)*Q1[i-m]+Q2[i-m]):
     od:
     y[i] := factor(-upper_term/((r+i)*(r+i-1)+Q1[0]*(r+i)+Q2[0])):
 od;

 i := 'i':  m := 'm': upper_term := 'upper_term':

# COMPLEX ROOTS OF THE INDICIAL EQUATION
# REAL SOLUTIONS DERIVED FROM THE COMPLEX SOLUTIONS
```

```
 Cond1 := Im(r1):
 if Cond1 <> 0 then
     sol1:=0:
     for m from 0 to Order-1 do
         sol1 := sol1 + y[m]*indep_var^m :
     od:
     K1:= subs(I=0,sol1):
     K2 :=normal(subs(I=1,sol1-K1)):
     sol1 := indep_var^Re(r1)* (cos(Im(r1)*ln(indep_var))*K1-
             sin(Im(r1)*ln(indep_var))*K2 ):
     sol2 := indep_var^Re(r1)* (cos(Im(r1)*ln(indep_var))*K2+
             sin(Im(r1)*ln(indep_var))*K1 ):
 else
     sol1 := 0:
     for m from 0 to Order-1 do
         sol1 := sol1 + y[m]*indep_var^m :
     od:
     sol1:=indep_var^r*sol1:
# y := 'y':

# SOLUTION CORRESPONDING TO r=r2 (THREE CASES DEPENDING
# ON THE DIFFERENCEBETWEEN r1 & r2
# ************************************************************************

     Cond1 := type((r1-r2),integer):

# CASE 1:  r1-r2 IS NOT AN INTEGER AND r1 NOT EQUALS r2
# ******************************************************

# FIND COEFFICIENTS y[i] THROUGH RECURRENCE RELATION

     if r1<>r2 and Cond1=false then
          r := r2:
          y[0]:=1:
          for i from 1 to Order do
               upper_term := 0:
               for m from 0 to i-1 do
                   upper_term := upper_term + y[m]*((r+m)
                                          *Q1[i-m]+Q2[i-m]):
```

```
            od;
            y[i] := factor(-upper_term/((r+i)*(r+i-1)
                                +Q1[0]*(r+i)+Q2[0])):
        od:
        i := 'i':
        m := 'm':

        sol2 := 0:
        for m from 0 to Order-1 do
            sol2 := sol2 + y[m]*indep_var^m :
        od:
        sol2 := indep_var^r*sol2:
        m := 'm':  y := 'y':
    fi:

# CASE 2: r1 AND r2 ARE EQUAL
#*****************************

# SECOND SOLUTION IS OF THE FORM
# ln(x)*sol1+sum(y[i]x^(i+r1),i=1..Order)
# WHERE x IS INDEPENDENT VARIABLE.
# FIND COEFFICIENTS y[i] THROUGH RECURRENCE RELATION

    if r1 = r2 then
        c[1]:=normal(-(2*y[1]+Q1[1]*y[0])/(r1*(r1+1)
                            +Q1[0]*(r1+1)+Q2[0])):
        for n from 2 to Order do
            upper_term := 0:
            lower_term := (r1+n)*(r1+n-1) + Q1[0]*(r1+n) + Q2[0];
            for j from 1 to n do
                upper_term := upper_term + Q1[j]*y[n-j];
            od:
            upper_term2 := 0:   j := 'j':
            for j from 1 to n-1 do
                upper_term2:= upper_term2+((r1+j)*Q1[n-j]
                                        +Q2[n-j])*c[j];
            od:
            c[n] := normal((-2*n*y[n]-upper_term-upper_term2)
                                                /lower_term);
        od;
```

```
        sol2 := 0;

        for m from 1 to Order-1 do
            sol2 := sol2 + c[m]*indep_var^m :
        od:
        sol2 := sol1*ln(indep_var)+indep_var^r*sol2:
    fi;
#
# CASE 3: r1 - r2 IS AN INTEGER
#*********************************

# SECOND SOLUTION IS OF THE FORM
# K1*ln(x)*sol1+sum(y[i]x^(i+r2),i=0..Order)
# WHERE x IS INDEPENDENT VARIABLE.
# FIND COEFFICIENTS y[i] THROUGH RECURRENCE RELATION

    if r1<>r2 and Cond1=true then
        if Order < (r1-r2) then
            lprint('Increase the Order at least upto',(r1-r2+2));
            lprint('and rerun the routine');
            RETURN('NULL'):
        fi;
        r := r2:
        N:= r1-r2;
        c[0] := 1:
        c[N] := 0:
        for n from 1 to N-1 do
            upper_term := 0:
            lower_term := (r2+n)*(r2+n-1) + Q1[0]*(r2+n) + Q2[0];
            for j from 0 to n-1 do
                upper_term := upper_term + ((r2+j)*Q1[n-j]
                                           + Q2[n-j])*c[j];
            od:
            c[n] :=  - upper_term / lower_term;
        od;
        upper_term := 0:   j := 'j':
        for j from 0 to N-1 do
            upper_term:= upper_term+((r2+j)*Q1[N-j]+Q2[N-j])*c[j];
        od:
```

```
            K1 := - upper_term / N;
            n:='n':
            for n from N+1 to Order do
                upper_term := 0:   j := 'j':
                lower_term := (r2+n)*(r2+n-1) + Q1[0]*(r2+n) + Q2[0];
                for j from 1 to n-N do
                    upper_term := upper_term + Q1[j]*y[n-N-j];
                od:
                upper_term2 := 0:    j := 'j':
                for j from 0 to n-1 do
                    upper_term2 := upper_term2+((r2+j)*Q1[n-j]
                                                +Q2[n-j])*c[j];
                od;
                c[n] := (-K1*((2*n-N)*y[n-N]+upper_term)
                          -upper_term2)/lower_term;
            od;

            sol2 := 0:
            for m from 0 to Order-1 do
                sol2 := sol2 + c[m]*indep_var^m :
            od:
            sol2 := K1*sol1*ln(indep_var)+indep_var^r*sol2:

        fi:
    fi:

# FROBENIUS SERIES SOLUTION

    sol := _C1*sol1+_C2*sol2:

# RETURN THE ROOTS OF THE INDICIAL EQUATION
# TO THE THIRD PARAMERTER (OPTIONAL)

        if nargs > 2 and type(args[3],name) then
            assign(args[3], [r1,r2]);
        elif nargs > 2 then
            ERROR('Can not assign a value to', args[3])
        fi;
    sol;
end:
```

```
#*******************************#
#     PARTICULAR INTEGRAL       #
#      BY SERIES METHOD         #
#        (Nth ORDER)            #
#*******************************#

interface(verboseproc=0):
# PARTICULAR SOLUTION
# LET ORIGIN BE THE REGULAR SINGULAR POINT OF Ly = F THEN
# IF R.H.S. OF DIFFERENTIAL EQUATION HAS A TAYLOR SERIES EXPANSION
# ABOUT x = 0, THEN THIS PROCEDURE OBTAINS THE PARTICULAR INTEGRAL
# OF Nth ORDER DIFFERENTIAL EQUATION BY SERIES METHOD, UNIQUELY.

Pi_Series_N := proc(deqn,var)
             local indep_var, yp, L_H_S, R_H_S, Q, gamm_a, beta,
             alpha, b, y, i, R, a, de1, fx,
             sol_set, sl, sr, ord:

# deqn: pN(x)*diff(y(x),x$2)+...+p1(x)*diff(y(x),x)+p0(x)*y(x)=F(x)
# var : DEPENDENT VARIABLE OF INDEPENDENT VARIABLE, e.g. y(x)

 indep_var := op(var):  # EXTRACT INDEPENDENT VARIABLE, x

# CALCULATE ORDER OF THE ODE BY INCREMENTING ORDER UNTIL IT
# REACHES THE HIGHEST DIFFERENTIATION IN THE ODE.

 for ord from 0 while has(lhs(deqn),diff(var,indep_var$ord+1 )) do
 od:

# EXTRACT COEFFICIENTS P(i)

 R := subs( seq(diff(var,indep_var$(ord-i+1))=D[ord-i+1],i=1..ord)
                                                  ,lhs(deqn)):

 i:= 'i':
 for i from 1 to ord-1 do
     Q[i] := convert(series((coeff(R,D[ord-i])/coeff(R,D[ord]))
                 *(indep_var^i),indep_var,Order+2*ord),polynom);
     b[i] := ldegree(Q[i],indep_var):
 od:
```

```
 Q[ord] := convert(series((coeff(R,var)/coeff(R,D[ord]))
                *(indep_var^ord),indep_var,Order+2*ord),polynom);
 b[ord] := ldegree(Q[ord],indep_var):


# R.H.S. OF DIFFERENTIAL EQUATION
# f(x) = x^N*F(x)/pN(x)

 fx := indep_var^ord*rhs(deqn)/coeff(R,D[ord]);

 R_H_S := convert(series(fx,indep_var,Order+2*ord),polynom);
 a := ldegree(R_H_S,indep_var):
 i:= 'i':

# FIND alpha AND beta AS DEFINED IN THEOREM FOR THIS METHOD
# Order IS SAME AS 'D' USED IN THEOREM

 alpha := a-min(seq(b[i],i=1..ord));
 beta := Order - alpha + a;
 i:= 'i':

# CHECK THE VALIDITY OF THE THEOREM

 if alpha > a  then
     lprint('Method not valid for this ODE.');
     lprint('Use either Power Series method or Variation
                                        of Parameters');
     RETURN('NULL'):
 fi;

# SERIES EXPANSION OF THE COEFFECIENTS Q[i] NEED ONLY
# TO BE CONTINUED TO DEGREE gamma = Order-2*alpha+a.
# IF Order+4 (USED BEFORE) IS LESS THAN gamma, THEN
# INCREASE THE ORDER OF THE SERIES EXPANSION UPTO gamma.

 gamm_a:= (Order -2*alpha +a ):
 for i from 1 to ord do
     if ((Order+2*ord) < gamm_a+ord )  then
         Q[i] := convert(series((coeff(R,D[ord-i])/coeff(R,D[ord]))
```

```
                *(indep_var^i),indep_var,gamm_a+ord),polynom);
    fi;
 od;
 i:= 'i':

 if ((Order+2*ord) < gamm_a+ord )  then
      Q[ord] := convert(series((coeff(R,var)/coeff(R,D[ord]))
                *(indep_var^ord),indep_var,gamm_a+ord),polynom);

 fi:
# SERIES EXPANSION OF RHS OF THE DIFFERENTIAL EQUATION NEED ONLY
# TO BE CONTINUED TO DEGREE beta. IF Order+4 (USED BEFORE) IS
# LESS THAN beta, THEN INCREASE THE ORDER OF THE SERIES
# EXPANSION UPTO beta.

 if ( Order+2*ord <= beta+ord )  then
       R_H_S := convert(series(fx,indep_var,beta+ord),polynom);
 fi;

# PUT THE EQUATION IN STANDARD FORM FOR THE METHOD OF FROBENIUS.

 de1 := indep_var^ord*diff(var,indep_var$ord)+sum(indep_var^(ord-i)
        *Q[i]*diff(var,indep_var$(ord-i)),i=1..ord-1) + Q[ord]*var:
 i:='i':

 yp := sum(y[i]*indep_var^i, i=alpha..Order);
 i:='i':

# SUBSTITUTE yp IN THE LHS OF THE ODE IN STANDARD FORM AND COLLECT
# IN THE INDEPENDENT VARIABLE

 L_H_S := collect(expand(subs(ln(indep_var)=0,subs(var=yp,de1)))
                                                  ,indep_var);

# COMPARE THE LIKE POWERS OF THE INDEPENDENT VARIABLE ON BOTH
# SIDES TO FIND y[i] I'    RTICULAR INTEGRAL

 for i from a to beta do
     sl[i]:=coeff(L_H_S,indep_var,i):
     sr[i]:=coeff(R_H_S,indep_var,i):
```

```
 od:
 i:='i':

# SOLVE THE SYSTEM OF EQUATIONS FOR y[i]

 sol_set:=solve({(sl[i]=sr[i])$i=a..beta},{y[i]$i=alpha..Order}):
 assign(sol_set):

# IF THE GIVEN DIFFERENTIAL EQUATION 'deqn' DOES NOT SATISFY THE
# CONDITIONS OF THE THEOREM USED IN THIS PROGRAM, THEN USE
# VARIATION OF PARAMETERS.

 if has({seq(y[i],i=alpha..Order)},'y') then
        lprint('Method not valid for this ODE.');
        lprint('Use Variation of Parameters');
        RETURN();
 fi;

# THE PARTICULAR INTEGRAL

 eval(yp);
end:
```

```
#*******************************#
#     PARTICULAR INTEGRAL       #
#      BY SERIES METHOD         #
#        (SECOND ORDER)         #
#*******************************#

interface(verboseproc=0):
# PARTICULAR SOLUTION
# LET ORIGIN BE THE REGULAR SINGULAR POINT OF Ly = F THEN
# IF R.H.S. OF DIFFERENTIAL EQUATION HAS A TAYLOR SERIES EXPANSION
# ABOUT x = 0, THEN THIS PROCEDURE OBTAINS THE PARTICULAR INTEGRAL
# OF SECOND ORDER DIFFERENTIAL EQUATION BY SERIES METHOD, UNIQUELY.

Pi_Series_2 := proc(deqn,var,r,sol)
             local indep_var, yp, L_H_S, R_H_S, Q, gamm_a, beta,
             alpha, b, k, y, i, R, a, r1, r2, A, de1, fx, Cond1,
             sol_set, sl, sr, y1, y2, B,Y2,YY:

# deqn : p2(x)*diff(y(x),x$2)+p1(x)*diff(y(x),x)+p0(x)*y(x) = F(x)
# var  : DEPENDENT VARIABLE OF INDEPENDENT VARIABLE, e.g. y(x)
# r    : LIST OF r1 AND  r2, i.e. [r1,r2]
# sol  : SOLUTION FROM THE FROBENIUS PROCEDURE

 indep_var := op(var):  # EXTRACT INDEPENDENT VARIABLE, x

# EXTRACT COEFFICIENTS P(i)

 R := subs( seq(diff(var,indep_var$(3-i))=D[3-i],i=1..2),lhs(deqn)):
 i:= 'i':
 Q[1] := convert(series(indep_var*coeff(R,D[1])/coeff(R,D[2])
                    ,indep_var,Order+4),polynom);
 b[1] := ldegree(Q[1],indep_var):

 Q[2] := convert(series((indep_var^2)*coeff(R,var)/coeff(R,D[2])
                    ,indep_var,Order+4),polynom);
 b[2] := ldegree(Q[2],indep_var):

# R.H.S. OF DIFFERENTIAL EQUATION
# f(x) = x^2*F(x)/p2(x)
```

```
 fx := indep_var^2*rhs(deqn)/coeff(R,D[2]);

 R_H_S := convert(series(fx,indep_var,Order+4),polynom);
 a := ldegree(R_H_S,indep_var):

# FIND alpha AND beta AS DEFINED IN THEOREM FOR THIS METHOD
# Order IS SAME AS 'D' USED IN THEOREM

 alpha := a-min(seq(b[k],k=1..2));
 beta := Order - alpha + a;

# CHECK THE VALIDITY OF THE THEOREM

 if alpha > a  then
     lprint('Method not valid for this ODE.');
     lprint('Use either Power Series method or Variation
                                          of Parameters');
     RETURN('NULL'):
 fi;

# SERIES EXPANSION OF THE COEFFECIENTS Q[k] NEED ONLY
# TO BE CONTINUED TO DEGREE gamma = Order-2*alpha+a.
# IF Order+4 (USED BEFORE) IS LESS THAN gamma, THEN
# INCREASE THE ORDER OF THE SERIES EXPANSION UPTO gamma.

     gamm_a:= (Order -2*alpha +a ):
     if ((Order+4) <= gamm_a+2 )  then
        Q[1] := convert(series(indep_var*coeff(R,D[1])/coeff(R,D[2])
                     ,indep_var,gamm_a+2),polynom);

        Q[2] := convert(series((indep_var^2)*coeff(R,var)
                /coeff(R,D[2]),indep_var,gamm_a+2),polynom);
     fi;

# SERIES EXPANSION OF RHS OF THE DIFFERENTIAL EQUATION NEED ONLY
# TO BE CONTINUED TO DEGREE beta. IF Order+4 (USED BEFORE) IS
# LESS THAN beta, THEN INCREASE THE ORDER OF THE SERIES
# EXPANSION UPTO beta.

 if ( Order+4 <= beta+2 )  then
```

```
        R_H_S := convert(series(fx,indep_var,beta+2),polynom);
  fi;

 # PUT THE EQUATION IN STANDARD FORM FOR THE METHOD OF FROBENIUS.

  de1 := indep_var^2*diff(var,indep_var$2) + indep_var*Q[1]*
                              diff(var,indep_var) + Q[2]*var:

 # r1 AND r2 ARE FIRST AND SECOND ROOTS OF THE INDICIAL EQUATION.

  r1 := op(1,r);     r2:= op(2,r);
  Cond1 := type((r1-r2),integer):

 # CASE 1:  r1-r2 IS NOT AN INTEGER AND r1 NOT EQUALS r2
 # ******************************************************

  if r1<>r2 and Cond1=false then      ## begin Case 1 'if'

 #USE THE VALUE OF 'alpha' AS LOWER LIMIT FOR THE PARTICULAR INTEGRAL

      yp := sum(y[i]*indep_var^i, i=alpha..Order);

 # SUBSTITUTE yp IN THE LHS OF THE ODE IN STANDARD FORM AND COLLECT
 # IN THE INDEPENDENT VARIABLE

      L_H_S := collect(expand(subs(ln(indep_var)=0,subs(var=yp,de1))
                        ),indep_var) ;

 # COMPARE THE LIKE POWERS OF THE INDEPENDENT VARIABLE ON BOTH
 # SIDES TO FIND y[i] IN PARTICULAR INTEGRAL

      for i from a to beta do
          sl[i]:=coeff(L_H_S,indep_var,i):
          sr[i]:=coeff(R_H_S,indep_var,i):
      od:
      i:='i':

 # SOLVE THE SYSTEM OF EQUATIONS FOR y[i]

      sol_set:=solve({(sl[i]=sr[i])$i=a..beta}
```

```
                    ,{y[i]$i=alpha..Order}):
     assign(sol_set):
 fi;                              ## end Case 1 'if'

# CASE 2: r1 AND r2 ARE EQUAL
#*****************************

 if r1 = r2 then                              ## begin Case 2 'if'

# TO COMPLETE THE CALCULATIONS, ORDER MUST BE GREATER THAN r1.

    if Order < r1 and type(r1,integer) then
       lprint('Increase the Order at least upto',(r1+2));
       lprint('and rerun the routine');
       RETURN('NULL'):
    fi;

# y1 AND y2 ARE FIRST AND SECOND FROBENIUS SERIES SOLUTIONS
# RESPECTIVELY. WHERE y2 = ln(x)*y1 + Y2;

    y1 := subs(_C1=1, _C2=0, sol);
    y2 := collect(subs(_C1=0, _C2=1, sol),ln(indep_var));
    Y2 := simplify(y2 - ln(indep_var)*y1);

# IF LOWEST DEGREE OF THE RHS 'a' IS GREATER THAN r1, THEN RHS OF
# THE EQUATION DOES NOT HAVE x^r1 TERM, HENCE THERE IS NO NEED TO
# INCLUDE THE LOGARITHMIC TERM IN THE PARTICULAR INTEGRAL.

    if a > r1 then A:=0: fi:

#USE THE VALUE OF 'alpha' AS LOWER LIMIT FOR THE PARTICULAR INTEGRAL

    yp:= ln(indep_var)*A*y1 + A*ln(indep_var)*ln(indep_var)*y1
         +2*A*ln(indep_var)*Y2+sum(y[i]*indep_var^i,i=alpha..Order);

# SUBSTITUTE yp IN THE LHS OF THE ODE IN STANDARD FORM AND COLLECT
# IN THE INDEPENDENT VARIABLE

     L_H_S := collect(subs(ln(indep_var)=0,expand(subs(var=yp,de1)))
                      ,indep_var) ;
```

```
# COMPARE THE LIKE POWERS OF THE INDEPENDENT VARIABLE ON BOTH
# SIDES TO FIND y[i] IN PARTICULAR INTEGRAL

     for i from a to beta do
         sl[i]:=coeff(L_H_S,indep_var,i);
         sr[i]:=coeff(R_H_S,indep_var,i);
     od;
     i:='i':

# SOLVE THE SYSTEM OF EQUATIONS FOR y[i] AND 'A'

     y[r1] := 0:

     if a > r1 then
           sol_set := solve({(sl[i]=sr[i])$i=a..beta},
                      {y[i]$i=alpha..r1-1,y[i]$i=r1+1..Order});
           assign(sol_set);
     else
           sol_set := solve({(sl[i]=sr[i])$i=a..beta},
                     {A,y[i]$i=alpha..r1-1,y[i]$i=r1+1..Order});
           assign(sol_set);
     fi:
     yp:= collect(simplify(convert(series(eval(yp),indep_var
                              ,Order-1),polynom)),indep_var);

 fi;                              ## end of Case 2 'if'

# CASE 3: r1 - r2 IS AN INTEGER
#*********************************

 if r1<>r2 and Cond1=true then          ## begin Case 3 'if'

# TO COMPLETE THE CALCULATIONS, ORDER MUST BE GREATER THAN r1.

    if Order < r1 and type(r1,integer) then
       lprint('Increase the Order at least upto',(r1+2));
       lprint('and rerun the routine');
       RETURN('NULL'):
    fi;
```

```
# y1 AND y2 ARE FIRST AND SECOND FROBENIUS SERIES SOLUTIONS.
# y2 = K*ln(x)*y1 + Y2, WHERE 'K' MAY OR MAY NOT BE ZERO.
# YY = K*ln(x)

     y1 := subs(_C1=1, _C2=0, sol);
     y2 := collect(subs(_C1=0, _C2=1, sol),ln(indep_var));
     coeff(",ln(indep_var)):
     Y2 := y2 - ln(indep_var)*";
     YY := y2 - Y2:

#IF LOWEST DEGREE OF THE RHS 'a' IS GREATER THAN r2, THEN RHS OF THE
# EQUATION DOES NOT HAVE x^r2 TERM, HENCE WE CAN TAKE 'B' = 0;

     if a > r2 then B:= 0: fi:

#IF LOWEST DEGREE OF THE RHS 'a' IS GREATER THAN r1, THEN RHS OF THE
# EQUATION DOES NOT HAVE x^r1 AND x^r2 TERM, HENCE THERE IS NO NEED
# TO INCLUDE THE LOGARITHMIC TERM IN THE PARTICULAR INTEGRAL.

     if a > r1 then A:=0: B:= 0: fi:

#USE THE VALUE OF 'alpha' AS LOWER LIMIT FOR THE PARTICULAR INTEGRAL

     yp:= ln(indep_var)*A*y1+B*ln(indep_var)*YY+2*B*ln(indep_var)*Y2
                          + sum(y[i]*indep_var^i, i=alpha..Order);

# SUBSTITUTE yp IN THE LHS OF THE ODE IN STANDARD FORM AND COLLECT
# IN THE INDEPENDENT VARIABLE

     L_H_S := collect(subs(ln(indep_var)=0,expand(subs(var=yp,de1)))
                      ,indep_var) ;

# COMPARE THE LIKE POWERS OF THE INDEPENDENT VARIABLE ON BOTH
# SIDES TO FIND y[i] IN PARTICULAR INTEGRAL

     y[r1] := 0:
     y[r2] := 0:

     for i from a to beta do
```

```
        sl[i]:=coeff(L_H_S,indep_var,i);
        sr[i]:=coeff(R_H_S,indep_var,i);
    od;
    i:='i':

# SOLVE THE SYSTEM OF EQUATIONS FOR y[i], 'A' AND 'B'

    if a > r1 then
        sol_set := solve({(sl[i]=sr[i])$i=a..beta},
                  {y[i]$i=alpha..r1-1,y[i]$i=r1+1..Order});
        assign(sol_set);
    elif  a <= r1 and a > r2 then
        sol_set := solve({(sl[i]=sr[i])$i=a..beta},
                 {A,y[i]$i=alpha..r1-1,y[i]$i=r1+1..Order});
        assign(sol_set);
    else
        sol_set := solve({(sl[i]=sr[i])$i=a..beta},
                 {A,B,y[i]$i=alpha..r2-1,y[i]$i=r2+1..r1-1,
                  y[i]$i=r1+1..Order});
        assign(sol_set);
    fi:
    yp:= collect(simplify(convert(series(eval(yp),indep_var
                 ,Order-1),polynom)),indep_var);
fi;                                    ## end of Case 3 'if'

# IF THE GIVEN DIFFERENTIAL EQUATION 'deqn' DOES NOT SATISFY THE
# CONDITIONS OF THE THEOREM USED IN THIS PROGRAM, THEN USE
# VARIATION OF PARAMETERS.

    if has({seq(y[i],i=alpha..Order)},'y') then
       lprint('Method not valid for this ODE.');
       lprint('Use Variation of Parameters');
       RETURN();
    fi;

# THE PARTICULAR INTEGRAL

eval(yp);
end:
```

```
#*******************************#
#     PARTICULAR INTEGRAL       #
# BY VARIATION OF PARAMETERS    #
#          (Nth ORDER)          #
#*******************************#

interface(verboseproc=0):

# PROCEDURE FOR THE METHOD OF VARIATION OF PARAMETER FOR Nth ORDER
# ORDINARY DIFFERENTIAL EQUATION.

# deqn: pN(x)*diff(y(x),x$n)+...+p1(x)*diff(y(x),x)+p0(x)*y(x)=f(x)
# WRITTEN AS
# diff(y(x),x$n)+...+p1/pN*diff(y(x),x)+p0/pN*y(x) = f(x)/pN

Variation_of_Parameters := proc(sol,g,ord,indep_var)
                              local W,i,k,Wr,Wr_inv,u,g_ser:

# sol   : 'N' LINEARLY INDEPENDENT SOLUTIONS OF 'deqn' [with f(x) =0]
# g     : f(x)/pN
# ord   : ORDER OF THE ODE
# indep_var : INDEPENDENT VARIABLE

# SERIES OF g = f(x)/pN

 g_ser := convert(series(g,indep_var,Order+ord),polynom):

# WRONSKIAN OF 'N' LINEARLY INDEPENDENT SOLUTIONS

 linalg[Wronskian](linalg[vector]([seq(sol[i],i=1..ord)])
                                              ,indep_var):
 W:= convert(simplify(linalg[det]('')),polynom):
 series(1/W,indep_var,Order+ord):
 Wr_inv := convert('',polynom):

# WRONSKIAN OF SUBSETS OF THE (N-1) LINEARLY INDEPENDENT SOLUTIONS.

 for k from 1 to ord do
    linalg[Wronskian](linalg[vector]([seq(sol[i],i=1..(k-1)),
                     seq(sol[i],i=(k+1)..ord)]),indep_var):
```

```
    Wr[k]:=((-1)^(ord-k))*convert(simplify(linalg[det](")),polynom):
    u[k] := int(expand(Wr[k]*g_ser*Wr_inv),indep_var):
 od:
 k := 'k':
 expand(simplify(sum(u[k]*sol[k],k=1..ord)));
 subs(0=0,series(map(factor,"),indep_var,Order-1)):
 ":
end:
```

References

[1] Abramowitz, M. and Stegun, I. 1970. *Handbook of Mathematical Functions*, Dover.

[2] Barton, D. and Fitch, J.P. 1974. CAMAL: the Cambridge Algebra System, *SIGSAM Bull.*, **8**, 17-23.

[3] Batchelor, G.K. 1967. *An introduction to fluid mechanics*, Cambridge University Press.

[4] Batchelor, G.K. 1974. Transport properties of 2-phase material, with random structure, *Ann. R. Fluid*, **6**, 227-255.

[5] Björnståhl, Y. 1935. *Physics*, **6**, 257.

[6] Björnståhl, Y. and Snellman, K.O. 1937. *Kolloid-Z*, **78**, 258.

[7] Björnståhl, Y. and Snellman, K.O. 1939. *Kolloid-Z*, **86**, 223.

[8] Boyce, W.E. and Ecker, J.G., 1992. Revitalising calculus with a computer algebra system, *Siam News*, January 12.

[9] Char, B.W., Geddes, K.O., Gentleman, W.M. and Gonnet, G.H. 1983. The design of Maple: A Compact, Portable and Powerful Computer Algebra System, *in* J.A. vanHulzen (ed.), *Computer Algebra- (Proceedings of EUROCAL' 83), lecture notes in Computer Science*, **162**, Springer, 101-115.

[10] Corless, R.M. and Jeffrey, D.J. 1988. Stress moments of nearly touching spheres in low Reynolds number flow,*J. Appl. Math. Phys.*,**39**, 874-884.

[11] da Andrade, E.N. and Dodd, C. 1939. Effect of electric field on the viscosity of liquids, *Nature*, **143**, 26-27.

[12] da Andrade, E.N. and Dodd, C. 1946. The effect of electric field on the viscosity of liquids, *Proc. Roy. Soc.*, **A187**, 296-337.

[13] da Andrade, E.N. and Dodd, C. 1951. The effect of electric field on the viscosity of liquids, *Proc. Roy. Soc.*, **A204**, 449-464.

[14] da Andrade, E.N. and Hart, J. 1954. The effect of electric field on the viscosity of liquids, *Proc. Roy. Soc.*, **A225**, 463-472.

[15] Davenport, J.H., Siret, Y. and Tournier, E. 1993. *Computer Algebra sytems snd algorithms for algebraic computation*, Academic Press.

[16] Delaunay, C.E., 1867. *Théorie du Mouvement de la Lune*, Vol. 1 and 2, Mallet-Bachelier, Paris.

[17] DellaDora, J. and Tournier, E. 1981. Formal solutions of differential equations in the neighborhood of singular points (Regular and Irregular), *Proc. 1981 ACM Symp. on Symbolic and algebraic computation*, 25-29.

[18] Deprit, A., Henrad, J. and Rom, A., 1970. Lunar Ephemeris: Delaunay's theory revisited, *Science*, **168**, 1569-1570.

[19] Duff, A.W., 1896. *Phys. Rev.*, **4**, 23.

[20] Durlofsky, L., Brady, J.F. and Bossis, G. 1987. Dynamic simulation of hydrodynamically interacting particles, *J. Fluid Mech.*, **180**, 21-49

[21] Durlofsky, L. and Brady, J.F. 1989. Dynamic simulation of bounded suspensions of hydrodynamically interacting particles, *J. Fluid Mech.*, **200**, 39-67.

[22] Edwards, Jr., C.H. and Penny, D.E. 1989. em Elementary Differential Equations with Applications, Prentice Hall Inc., NJ 567 p.

[23] Einstein, Albert 1956. *Investigations on the theory of the Brownian movement*, Dover, New York.

[24] Geddes, K.O., Czapor, S.R. and Labahn, G. 1992. *Algorithms for Computer Algebra*, Kluwer Academic Pub.

[25] Happel, J. and Brenner, H. 1973. *Low Reynolds number hydrodynamics*, Martinus Nijhoff.

[26] Harper, D., Wooff, C. and Hodgkinson, D. 1991. *A Guide to Computer Algebra Systems*, Wiley Pub.

[27] Hartsock, D.L., Novak, R.E. and Chaundi, G.J. 1991. ER fluid requirements for automotive devices, *J. Rheol.*, **35**, 1305.

[28] Heck, A. 1993. *Introduction to Maple*, Springer-Verlag.

[29] Ince, E.L. 1957. *Ordinary Differential Equations*, Longmans, Green and Co. Ltd. NY, 558 p.

[30] Jeffrey, D.J. 1982. Low-Reynolds-number flow between converging spheres. *Mathematika*, **29**, 58-66.

[31] Jeffrey, D.J. 1989a. Stresslet resistance functions for low Reynolds number flow using deforming spheres, *J. Appl. Math. Phys.*, **40**, 163-171.

[32] Jeffrey, D.J. 1989b. Higher-order corrections to the axisymmetric interactions of nearly touching spheres, *Phys. Fluids*, **A1**, 1740-1742.

[33] Jeffrey, D.J. 1991. The lubrication analysis for two spheres in a two-dimensional pure-straining motion, *Phys. Fluids*, **A3**, 1819-1821.

[34] Jeffrey, D.J. 1992. The calculation of the low Reynolds number resistance functions for two unequal spheres, *Phys. Fluids*, **A4**, 16-29.

[35] Jeffrey, D.J. and Acrivos, A. 1976. Rheological properties of suspensions of rigid particles, *AICHE J*, **22**, 417-432.

[36] Jeffrey, D.J. and Corless, R.M. 1988. Forces and stresslets for the axisymmetric motion of nearly touching unequal spheres, *PhysicoChemical Hydrodynamics*, **10**, 461-470.

[37] Jeffrey, D.J. and Onishi, Y. 1984. Calculation of the resistance and mobility functions for two unequal rigid spheres in low-Reynolds-number flow, *J. Fluid Mech.*, **139**, 261-290.

[38] Jenks, R. and Suter, R.S. 1992. *AXIOM, The Scientific Computation System*, Springer.

[39] Kahrimanian, H.G. 1953. Analytic differentiation by a digital computer. M.A. Thesis, Temple Univ, Philadelphia, Pennsylvania.

[40] Kim, S. and Mifflin, R.T. 1985. The resistance and mobility functions of two equal spheres in low-Reynolds-number flow, *Phys. Fluids*, **28**, 2033-2045.

[41] Kim, S. and Karrila, S.J. 1991. *Microhydrodynamics: Principles and selected applications*, Butterworth-Heinemann.

[42] Knuth, D.E. 1992. Two Notes on Notation, *Amer. Math. Monthly*, **May**, 403-422.

[43] Latta, G.E. and Hess, G.B. 1973. Potential flow past a sphere tangent to a plane. *The Physics of Fluids*, **16**, 974-976.

[44] Moon, P. and Spencer, D.E. 1961. *Field Theory Handbook*, Springer, New York.

[45] Newton, I. 1728. *Arithmetica Universalis*, 2nd ed. London.

[46] Nolan, J. 1953. Analytic differentiation on a digital computer. M.A. Thesis, M.I.T., Cambridge, Massachussets.

[47] O'Neill, M.E. and Stewartson, K., 1967. On the slow motion of a sphere parallel to a nearby plane wall, *J. Fluid Mech.*, **27**, 705-724.

[48] Stoutemyer, D.R. 1991. Crimes and Misdemeanors in the Computer Algebra Trade, *Notices of the AMS*, **38**, 778-785.

[49] Stoutemyer, D.R. and Rich, A. 1983. *muMATH-83 Reference Manual*, The Software House: Honolulu, Hawaii.

[50] Van dyke, M., 1974. Computer extension of perturbation series in fluid mechanics, *SIAM Jour. Appl. Maths.*, **28**, 720-734.

[51] Winslow, W. 1949. Induced fibration of suspensions, *J. Appl. Phys.*, **20**, 1137-1140.

[52] Wolfram, S. 1991. *Mathematica: A System for doing mathematics by Computer*, Addison Wesley, 2nd ed.